Institute for Visualization and Interactive Systems

University of Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Master's thesis Nr.

Investigation of methods for reducing calibration effort in RSSI-based indoor locating systems

Artem Bannikov

Course of Study:	INFOTECH
Examiner:	Prof. Dr. Albrecht Schmidt
Supervisor:	Michael Oberle, Research Fellow at Fraunhofer IPA
Commenced:	February 2, 2015
Completed:	August 4, 2015
CR-Classification:	H.4.2 – logistics, C.2.1 – wireless communication, C.2.4 – client/server, C.5.3 – portable devices (e.g., laptops, personal digital assistants), D.1.5, H.2.4 – relational databases

Acknowledgments

I would like to express my gratitude to **Fraunhofer IPA** for the highly interesting research topic and to **Prof. Dr. Albrecht Schmidt**, who has supported it. I was happy to research and to create something of my own.

I am immensely grateful to my supervisor **Michael Oberle** for his genuine interest in the topic, for reliable and effective feedback and for constructive and fruitful discussions that we had during the whole research period. I sincerely thank to **Norman Pohl** for the ideas he provided concerning the research, to **Yomna Abdelrahman** for her help with organizational issues as well as to **Florian Maier**, **Olha Meyer**, **Raphael Depuydt** and **Matthias Stöhr**, who have been involved on different stages of the research. Your constant willingness to help is really valuable.

I would also like to thank my family, the friends of mine and those people, who are not indifferent to what I am doing. I always feel your support and it is very important for me.

Abstract

Indoor locating systems (ILSs) for locating assets (e.g. work pieces or tools) in manufacturing enterprises increase transparency, reduce search times on the shop floor and help to optimize intra-logistics of the enterprises. However, despite the potential for efficiency gains the adoption of ILSs in small and medium enterprises (SMEs) is restrained by the associated efforts. For SMEs an ideal ILS would be based on a standard low-cost technology for decreasing hardware costs, offer a simple way of introduction and minimal operational effort. Therefore, the thesis investigates design and associated localization methods of a Bluetooth low energy (BLE)- and received signal strength indicator (RSSI)-based ILS that possesses simplified calibration procedure or even totally eliminates it and, thus, decreases efforts related to its initial deployment.

The thesis proposes two research concepts. The first concept is intended for simplification of the calibration procedure (in fact, acquisition of fingerprints) and leverages automatic movement that is available in industry (e.g. a conveyor or a robot) for this purpose. The second concept proposes an idea of spatial distribution of fixed nodes into emitters and receivers. For investigation of the named research concepts a BLE- and RSSI-based ILS has been implemented. As a localization approach, the core idea of the ring overlapping based on comparison of received signal strength indicators (ROCRSSI) range-free localization technique is used in the system.

Tests of the implemented ILS conducted in two test venues have shown that fixed nodes spatially distributed into emitters and receivers perform better than the merged ones (both in terms of localization accuracy and in terms of deviation of localization errors from their means). Preliminary calibration of the system with the help of an automatic movement (in this particular case, with the help of a line-following robot) has led to ambiguous results and demands additional investigations.

Contents

List of Figures

List of Tables

1 Introduction

The introduction gives a motivation for this research, poses its objectives and briefly describes the scope of the thesis (see Sections 1.1, 1.2 and 1.3 correspondingly). The introduction also provides an outline of the thesis's structure (see Section 1.4).

1.1 Motivation

Indoor locating systems (ILSs) deployed at manufacturing enterprises are able to improve their efficiency due to generally improved transparency of shop floor operations. Real world usage reports confirm the positive effect that manufacturing enterprises are able to achieve while using locating systems. However, despite the potential for efficiency gains the adoption of ILSs in small and medium enterprises (SMEs) is restrained by the associated efforts. For SMEs an ideal ILS would be based on a low-cost technology for decreasing hardware costs, offer a simple way of introduction and minimal operational effort. An ILS system design, in particular appropriate localization methods that consider all of the previously stated characteristics demand an investigation. Contributing to this investigation is the overall goal of this thesis.

1.2 Objectives

Among all the technologies that are used for the indoor localization, such as infrared (IR), ultrasound and camera-based, radio frequency (RF) is the most favorable due to portability, low cost and low power consumption of the used sensors. Among different approaches for the indoor wireless (RF) localization methods, received signal strength indicator (RSSI)-based triangulation that operates with the path-loss nature of RF signals is the most promising for building a low-cost system, since it can be applied only with few modifications of the base technology and with hardware of standard wireless technologies such as Bluetooth low energy (BLE) and Wi-Fi.

For RSSI-based ILSs simplicity in their introduction (in their initial deployment) implies that efforts associated with calibration, which is used by the RSSI-based ILSs in order to increase their localization accuracy, have to be eliminated or reduced to a minimum.

Therefore, the thesis aims at the investigation of an RSSI-based ILS design and the associated localization methods that reduce or even totally eliminate efforts involved in the calibration procedure while providing a localization accuracy that is sufficient for reducing search times on the shop floor. The accuracy of localization depends on use cases, however a sub-room level (i.e. localization within the accuracy of a corner of the room) is typically sufficient for most use cases.

1.3 Scope

The calibration procedure of an ILS is in most cases based on acquisition of fingerprints (spatially anchored RSSI values) within a shop floor, where the system is going to be applied. Whereas researchers try to reduce the system's calibration time (for instance, by reducing the number of fingerprints) and thus to decrease the cost of the system, there are no investigations that leverage automatic movements that are often available at manufacturing enterprises (e.g. conveyors or robots) for the purpose of preliminary calibration of the system. Calibration with the help of an automatic movement may provide the following advantages:

1. A vast amount of fingerprints can be acquired along the conveyor (or along the motion path of any other source of automatic movement), if its speed and frequency of the fingerprints' acquisition allow to put it into practice. Increase in number of fingerprints, in theory, improves localization accuracy of the system.
2. Usage of an automatic movement for the purpose of preliminary calibration of the system may decrease the system's deployment time and thus reduce its cost.

In this thesis the ring overlapping based on comparison of received signal strength indicators (ROCRSSI) range-free localization technique is going to be used as an underlying localization approach. All of the found investigations of ROCRSSI-based ILSs consider the systems' fixed nodes (non-mobile nodes with known coordinates for reference) to be emitters and receivers at the same time. This is not desirable for a low-cost system, since hardware that is able to emit and receive is typically more expensive. The efficiency of ROCRSSI-based ILSs operating with the fixed nodes that are spatially distributed into emitters and receivers has not yet been researched.

Based on the identified knowledge gaps, two research concepts are proposed. The first concept leverage an automatic movement for the purpose of preliminary calibration of the system. The second one considers idea of spatial distribution of the system's fixed nodes into emitters and receivers. This master's thesis represents an investigation of the two proposed concepts. For their investigation an ILS has been designed and implemented. The ILS provides several localization strategies that thoroughly cover the two proposed concepts. The strategies are evaluated and compared with each other by accuracy of localization estimates and by deviations of localization errors from their means.

1.4 Outline

The rest of the thesis is organized as follows:

Chapter 2 – Literature review: The chapter presents use cases of ILSs in manufacturing enterprises and outlines existing indoor localization methods. Based on the identified knowledge gaps two research concepts are proposed. In the end of the chapter research questions formulated on the basis of the two proposed concepts are stated. The research questions are intended to identify approaches for ILS design that minimize required calibration effort.

Chapter 3 – System design: In this chapter the design of the prototype ILS that is intended to address the research questions posed in the previous chapter is presented. Firstly, the prototype ILS's architecture is described. Secondly, the underlying localization algorithm and localization strategies formulated on its basis and covering the two research concepts proposed in the previous chapter are explained. Then, mathematical basis of the fingerprints' acquisition with the help of an automatic movement that is a core idea of one of the proposed research concepts is provided. In the end, an approach to the noise filtering is proposed and an optimized beacon configuration is discussed.

Chapter 4 – System implementation: This chapter explains, how the system design proposed in the previous chapter has been implemented. It describes the graphical user interface (GUI) of the client's Android application, explains main implementation logic both on the client and on the server sides, provides description of the server's database (DB) structure and covers the question of the client-server communication. The description is supported by figures and Unified Modeling Language (UML) diagrams.

Chapter 5 – System evaluation: Several sessions of experiments have been conducted. The chapter provides both overview of the conducted experiments (including their methodological sequence and description of the test venues) and detailed analysis of their results. The detailed analysis is intended for evaluation of ideas underlying the two research concepts proposed in Chapter 2 as well as for evaluation of the implemented RSSI filter. The evaluation is performed in terms of localization accuracy and in terms of deviation of localization errors from their means. The chapter provides also the mathematical basis for the evaluation.

Chapter 6 – Conclusions: This chapter summarizes the thesis and provides suggestions for possible extensions of the implemented ILS as well as suggestions for future research.

2 Literature review

The outline of this chapter is as follows: Section 2.1 provides use cases of ILSs in manufacturing enterprises and gives real world usage reports that confirm their potential; Section 2.2 outlines existing indoor localization methods and describes advantages of the ROCRSSI range-free localization technique originally proposed by Liu et al. [1], which core idea is used in this master's thesis; Section 2.3 summarizes identified knowledge gaps; Section 2.4 formulates two research concepts and poses research questions aiming at closing the identified knowledge gaps.

2.1 Use cases of ILSs in manufacturing enterprises

ILS solutions that operate in real time can be effective for different industries. Taking into consideration only primary localization function of such kind of systems the following use cases can be proposed. In semiconductor industry ILSs are used, for instance, for localization of wafer lots on the shop floor. Any shop floor may leverage an ILS for the purpose of work pieces' or tools' search. Cosmetic industry, for example, may consider to use an ILS for localization of tanks for the cosmetic production.

However, compound solutions could bring more efficient results. Some of the tags that could be attached to the sought objects possess sensors and buttons. As a compound solution, it is proposed to not only localize the tag attached to a sought object, but also leverage data that can be obtained from its sensors when an estimation of location is performed. For example, temperature, humidity and pressure sensors may monitor technological process which parameters are strictly determined. Buttons can be used for receiving a feedback from an operator, who has approached the sought object. For instance, in cosmetic industry press of one of the tag's buttons may indicate that a tank is empty and should be filled out.

Hawkins [2] has reviewed another compound solution (the system named Applied Smart-Move), where two technologies have been combined: real-time locating system (RTLS) and workflow automation software integrating the RTLS with the already existing manufacturing execution system (MES). The author states that at least 20% of an operator's time is expended in tracking and locating lots and interacting with MES for logging of their locations. However the MES does not prevent from an erroneously logged location or from a lost lot.

The proposed and reviewed Applied SmartMove system solves the problem outlined above. Whenever a lot is selected by the MES for its further processing, the RTLS locates the lot. A flashing light-emitting diode (LED) on the tag attached to the lot helps to identify the lot when operator comes closer to it. The tag's display informs operator about the destination tool for further processing of the lot. Besides the LED and the display, the tag is also provided with buttons that allow to convey simple notifications (such as process outcome or emergency situation) to the system. All this frees operator to log notifications and lots' locations manually into the MES, minimizes errors while logging and saves operator's time.

Together with already named advantages of the Applied SmartMove system, such as raise of operator's productivity and two-way communication *to* and *from* the operator, Hawkins [2] states also the following one: information about locations of lots may be used for optimization of decisions on the shop floor. For instance, lots' transition time can be reduced, if a system of priorities assigned to the lots according to their proximity is introduced.

In the review [2] Hawkins provides also some statistics from customers. According to the author, a logic manufacturer implemented RTLS has claimed about an 82% decrease in lot delivery time, a 13% betterment in on-time delivery and a 7% improvement in labor productivity. A big memory manufacturer that has eliminated necessity to manually log lot locations in the MES saved $900k per year in operator time. A large semiconductor foundry that has deployed the Applied's workflow technology in order to automate standard factory exception processing has resulted in an 18% factory throughput growth.

Successful statistics listed above and the outlined Applied SmartMove system itself show clearly that compound solutions involving other systems of an enterprise and leveraging not only location of the tag attached to a sought object, but also some concomitant data from it or establishing a communication interface both *to* and *from* operator should be considered and proposed as use cases for ILSs.

Since the literature confirms potential of ILSs, there have been many attempts at localization strategies, which are reviewed in the next section.

2.2 Review of indoor localization methods

Akeila et al. [3] state in their paper that among all the sensors that are used for the indoor localization, such as ultrasonic sensors and cameras, the RF ones are the most favorable due to their portability, low cost and low power consumption. The authors also state that one of the wireless (RF) technologies that is widely used for the indoor localization is Bluetooth due to its present widespread among mobile phones and portable electronic devices as well as due to its low cost. The new BLE technology included as a key technology into the Bluetooth Core Specification Version 4.0, which official adoption has been declared by the Bluetooth

Special Interest Group (SIG) in July 2010 [4], also provides an additional and core advantage: low power consumption.

Different approaches for the indoor wireless localization methods have been already proposed: angle of arrival (AOA), distance-based triangulation such as RSSI-based and time of arrival (TOA)-based methods and fingerprint-based positioning [5]. The RSSI-based triangulation operates with the principle of RF signal's attenuation with the distance from the emitter. This method is the most promising for building a low-cost system, since it can be applied only with few modifications of the base technology and with hardware of standard wireless technologies such as BLE and Wi-Fi. In comparison, as it is stated by Akeila et al. [3], usage of TOA for obtaining distances entails the following difficulties: despite of good accuracy, it demands some modifications to the actual Bluetooth specifications, as well as highly accurate systems for time measure.

Luo et al. [6] distinguish two primary categories of RSSI-based localization techniques: range-free localization techniques that do not require prior knowledge about the surrounding environment and the effect on signal strength, and range-based localization techniques that demand initial calibration of the system in order to know, how the signal propagates within the environment. Luo et al. [6] also distinguish two main subcategories of the range-based localization techniques: RSSI map-based and path-loss model-based algorithms.

The RSSI map-based algorithms work with a pre-built map of fingerprints (spatially anchored RSSIs). A typical example of the RSSI map-based algorithm is the k-nearest neighbor (kNN) training based algorithm described by Bahl and Padmanabhan [7].

At the heart of the RSSI path-loss model-based algorithms lies a model of the signal strength's regression with the distance (path-loss model). Some parameters of this model demand their calibration grounding on a set of RSSI values collected at different sample points of the test venue. Wang et al. [5] and Oguejiofor et al. [8] present two different path-loss models and approaches for their calibration.

Calibration of the system, either for building of a fingerprints' map or for adjustment of a path-loss model, is time-consuming. This increases the system's deployment time and hence the system's cost. Reducing this effort has been approached by reducing the number of fingerprints required for calibration [9], which aims at minimizing the manual work involved.

Another way of reducing the system's deployment time lies in the total elimination of the system's calibration procedure (range-free localization techniques). Some of the range-free localization techniques show good localization performance – for instance, the refined ROCRSSI range-free localization technique presented by Frattini et al. and denoted as ROCRSSI++ [10]. Luo et al. [6] have shown that even the original, not refined version of the ROCRSSI range-free localization technique shows good results in comparison with RSSI path-loss model-based and RSSI map-based algorithms. Moreover, Liu and Wu [11] have demonstrated in their research that the ROCRSSI range-free localization technique outperforms approximate point-in-triangulation test (APIT) (another range-free localization technique proposed by He et al. [12]) in terms of several characteristics, although (as it is stated by He et al. [12]) the latter performs best for randomly deployed wireless sensor networks (WSNs) in comparison with other range-free localization techniques, such as centroid localization, DV-HOP localization and amorphous localization. The ROCRSSI localization technique operates with several fixed nodes, which position within an indoor area is known.

2.3 Identified knowledge gaps

The following knowledge gaps have been identified in the field of indoor localization:

1. Whereas reducing the calibration effort has been approached by reducing the number of fingerprints [9] (minimization of the manual work involved), there are no researches that leverage automatic movements that are often available in industry (e.g. conveyors) for the purpose of simplifying of the required calibration procedure. Calibration with the help of an automatic movement may decrease the system's deployment time and thus reduce its cost. Besides, the fingerprints that are going to be acquired during the calibration procedure could be considered as additional reference points of the system. The increase in number of reference points may, in theory, improve localization accuracy of the system.
2. In this thesis the ROCRSSI range-free localization technique is going to be used as an underlying localization approach. While the ROCRSSI does not require calibration it has the disadvantage that it assumes that the fixed nodes are emitters and receivers at the same time. This is not desirable for a low-cost system, since hardware that is able to emit and receive is typically more expensive. The effectiveness of the ROCRSSI in the case of fixed nodes that are spatially distributed into emitters and receivers has not yet been researched. The spatial distribution of fixed nodes into emitters and receivers allows to reduce expenditures on the system, since separate emitting and receiving devices are typically cheaper than the hardware that is able to emit and receive simultaneously. In this master's thesis, simple BLE beacons serve as signals' emitters. The beacons are inexpensive. This allows to easily increase the number of the beacons, which itself, in theory, should favour a betterment of the localization accuracy of the system.

Investigation in the field of the identified knowledge gaps is intended to identify approaches that are able to reduce cost of the final ILS and/or improve its localization accuracy.

2.4 Research questions

This master's thesis aims at closing identified knowledge gaps that are summarized in Section 2.3. As a result, two research concepts are proposed. The first concept aims at leveraging automatic movement that is available in industry (e.g. a conveyor) for the purpose of simplifying of the required calibration procedure. The second concept uses range-free localization technique with fixed nodes that are spatially distributed into emitters and receivers. Based on these two concepts the following research questions have been posed:

1. What is an effective design for an algorithm that leverages automatic movements for the calibration purposes?
2. How does the proposed calibration procedure with the help of an automatic movement influence the localization accuracy of the system?
3. How does spatial distribution of fixed nodes into emitters and receivers influence the localization accuracy of the system, which localization approach is based on the core idea of the ROCRSSI range-free localization technique?
4. How does joint usage of the two proposed concepts influence the localization accuracy of the system?
5. How do the two proposed concepts for minimizing (or even total elimination of) calibration time perform in comparison?

The research questions are intended to identify approaches for ILS design that minimize required calibration effort while achieving an accuracy that is able to reduce search times on the shop floor.

3 System design

In this chapter design of the prototype ILS that is intended to address the research questions posed in the previous chapter is presented. The chapter is structured as follows: Section 3.1 describes architecture of the prototype ILS; Section 3.2 explains the ROCRSSI range-free localization technique [1] and its refinements (namely, the ROCRSSI+ [13] and the ROCRSSI++ [10]), which core idea is used as a localization approach of the prototype ILS; in Section 3.3 five localization strategies based on the ROCRSSI and covering the two research concepts proposed in the previous chapter are explained; Section 3.4 provides mathematical basis of the fingerprints' acquisition with the help of an automatic movement that is a core idea of one of the proposed research concepts; in Section 3.5 an approach to the noise filtering is given; in Section 3.6 aspects of an optimized beacon configuration are discussed.

3.1 Architecture

This section introduces the principle components that form the implemented prototype ILS and presents the block diagram of the system.

3.1.1 Components

In order to address the research questions stated in Section 2.4 a prototype ILS has been designed and implemented. Only standard hardware has been used in consistence with the overall design goal of a low-cost system. Therefore the system has been built on the basis of BLE technology. The BLE devices possess low cost and consume low power. The latter advantage implies less efforts for maintenance of hardware (in particular, for changing batteries) and, consequently, reduction of the system's operational costs. The principle components that form the implemented ILS are listed and briefly described below:

1. **Anchor:** BLE compatible Android smartphone serving as signals' receiver. Coordinates of anchors are known to the system and are used for implementation of the localization algorithm. In the client-server architecture of the system anchors play role of clients. They scan for available BLE devices and form scan results that are sent to the server's DB. The Android clients are also used for filling the server's DB with experiments' configuration data and for graphical representation of the currently conducted experiment.

Communication between the Android clients and the server is implemented with the help of web services. For the prototype ILS implemented in this thesis Motorola Moto G 2014 Android smartphones have been used as anchors of the system.

2. **Reference node:** BLE beacon serving as signals' emitter. Coordinates of reference nodes are known to the system and are used for implementation of the localization algorithm. Texas Instruments (TI) CC2541 SensorTags serve as reference nodes of the system.
3. **Tag:** standard BLE beacon that plays role of an asset (sought) beacon. The TI CC2541 SensorTag acts as a tag.
4. **Access point:** Wi-Fi router. The access point is utilized for establishing a local wireless network that is used for the client-server communication. The TP-LINK AC750 wireless router serves as an access point of the system.
5. **Server:** laptop with Wi-Fi adapter. The server is used for allocation of the web services and DB. On the server localization algorithm is performed. For the prototype ILS the server has been allocated on a Lenovo ThinkPad T430 laptop.
6. **Source of automatic movement:** automatic movement that is available in industry (e.g. conveyor). The source of automatic movement is used for the purpose of preliminary calibration of the system (namely, in the fingerprints' acquisition procedure). In order to test the implemented ILS in an environment, where conveyor or any other industrial source of automatic movement is absent (e.g. in a classroom), a line-following robot has been proposed and used as a source of automatic movement. The Pololu m3pi robot has been used for this purpose.

3.1.2 Block diagram

Components of the system that are used for depiction of the block diagram have been listed and briefly described in Section 3.1.1. The implemented ILS presenting a client-server architecture is shown in Figure 3.1.

The block diagram of the system that is depicted in Figure 3.1 represents the case when fixed nodes are spatially distributed into emitters (reference nodes) and receivers (anchors). As a source of automatic movement, a conveyor is illustrated. Along the conveyor acquired fingerprints are shown. Theoretical basis of the fingerprints' acquisition procedure is described in Section 3.4, whereas its GUI and logic are provided in Sections 4.2.1 and 4.2.3 correspondingly.

Every Android client (anchor) operating in the system scans for available BLE devices (reference nodes and a tag) and obtains their RSSIs. The obtained RSSIs are accumulated and filtered. After the filtering the client forms a scan result and sends it to the server's DB. Any Android client can also be used for configuration of experiments. Implementation of the scan

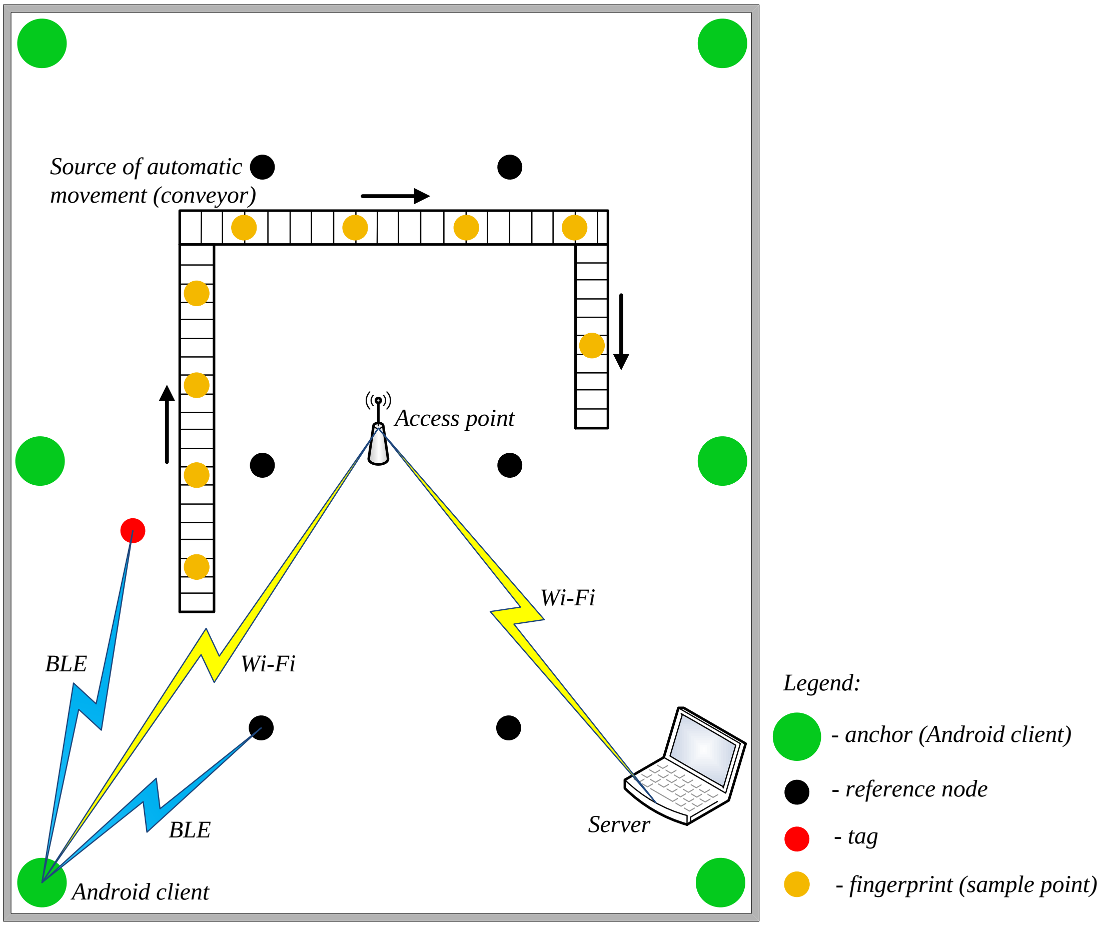


Figure 3.1: Block diagram of the system

procedure (together with getting RSSIs, filtering them for noise, formation of a scan result and sending the scan result to the server's DB) is described in Section 4.2.2. The implemented GUI that is responsible for the configuration of experiments is demonstrated in Section 4.2.1.

Every Android client is able to communicate with the server via the dedicated local Wi-Fi network. The communication is organized with the help of web services.

The server allocates web services and DB and performs localization algorithm using the current experiment's configuration data and scan results received from the clients. Implementation details of the client-server communication (in particular, its core logic, main scenarios and procedures) are described in Section 4.4. Description of structure of the DB is provided in Section 4.3.1.[1]

3.2 Localization algorithm

This section presents theoretical basis of the ROCRSSI range-free localization technique, which core idea is used in the implemented ILS. First of all, idea of original ROCRSSI [1] is described in detail. Then, its refinements (namely, the ROCRSSI+ [13] and the ROCRSSI++ [10]) are described. In the end, the grid-scan algorithm [12] that aims at calculation of the center of gravity (CoG) of the rings' intersection area found by the ROCRSSI is explained.

3.2.1 Original ROCRSSI range-free localization technique

As a localization approach, the core idea of the ROCRSSI range-free localization technique proposed by Liu et al. [1] is applied for the system. The ROCRSSI uses rings' intersection for the estimation of the tag's location. Liu et al. [1] consider that a tag (the authors name it as "sensor node") operates with a series of overlapping rings in order to narrow down possible area, where it is located. In the implemented ILS calculation of the rings' intersection area is done by the server. The calculation is based on measurements made by each anchor. Below the ROCRSSI proposed by Liu et al. [1] will be interpreted according to the architectural and hardware characteristics of the implemented ILS. However, its core idea stays unchanged.

Liu et al. [1] state that the core idea of the ROCRSSI is based on the assumption that along a certain direction the signal strength decreases monotonically with the distance. This assumption allows to predict, grounding on the received signal strengths, which from the senders is spatially farther or closer to the receiver.

Figure 3.2 shows three fixed nodes A, B and C and a tag T, which location has to be found. Every fixed node can emit and receive signals, the tag T only emits signals. The rings can be generated by comparison of signal strengths a fixed node receives from the tag T and from the rest of two fixed nodes. For instance, let us assume that the fixed node A receives the following signal strengths from the tag T and fixed nodes B and C respectively: $RSSI_{TA}$, $RSSI_{BA}$, $RSSI_{CA}$. The last subscript letter in the RSSI notations denotes the node that measured the signal strength and the first subscript letter denotes the node (or tag) emitted the signal. Let us also assume that the received signal strengths form the following inequalities: $RSSI_{CA} > RSSI_{TA} > RSSI_{BA}$. In that case the tag T most probably falls inside the ring with center at fixed node A, and inner and outer radiuses AC and AB respectively. In

[1] The data model diagram (DMD) that reflects the structure of the server's DB is depicted in Figure B.1 in appendices.

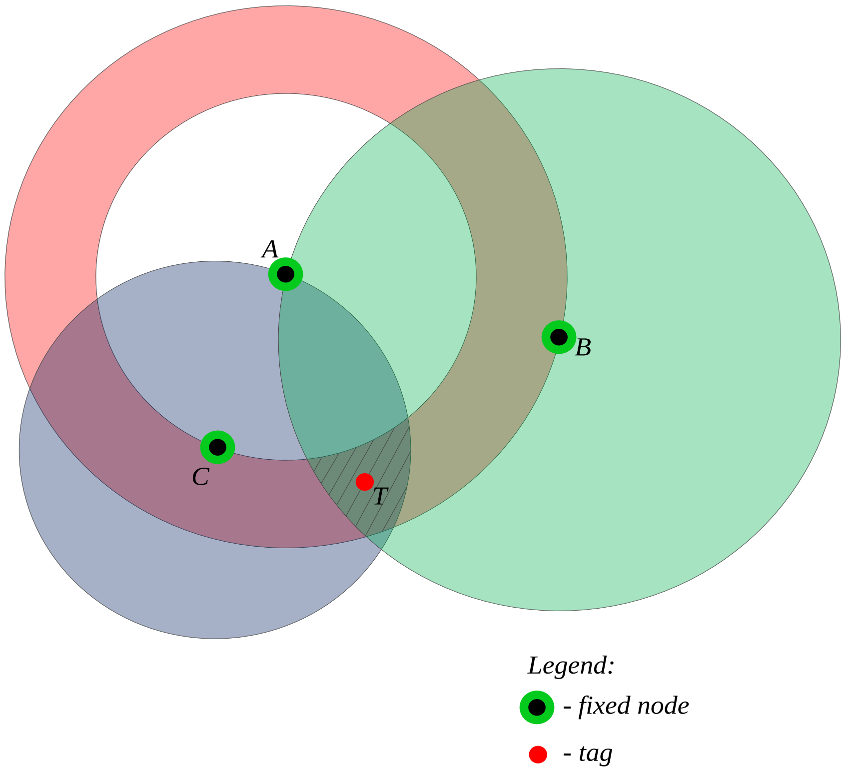


Figure 3.2: Core idea of ROCRSSI by the example of three fixed nodes A, B and C and a tag T

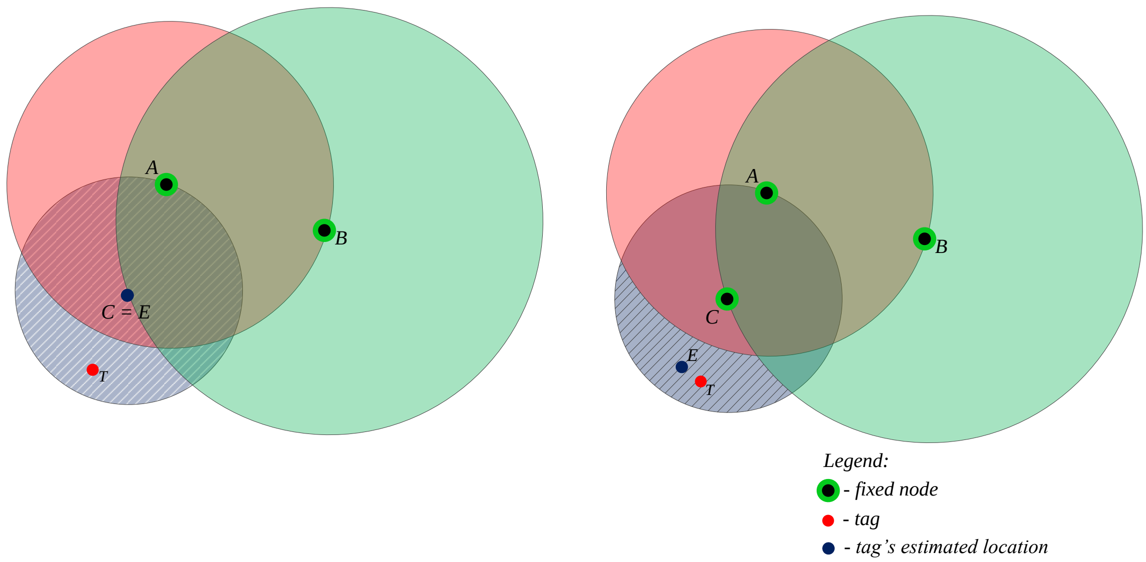
a similar way, let us assume that for the fixed nodes B and C the following inequalities are fulfilled: $RSSI_{TB} > RSSI_{AB} > RSSI_{CB}$ (then the tag T most probably falls inside the circle with center at fixed node B and radius BA) and $RSSI_{TC} > RSSI_{AC} > RSSI_{BC}$ (then the tag T most probably falls inside the circle with center at fixed node C and radius CA). As a result, the tag T can be located at the CoG of the hatched intersection of the ring and two circles. The ROCRSSI does not map the received signal strengths to the corresponding absolute distances between the senders and the receivers. Therefore the ROCRSSI is categorized as a range-free localization technique.

It should be mentioned here that surrounding environment is usually different in different directions. Zhou et al. [14] note that this causes irregularity of radio propagation in most environments. That is, different directions have different path-loss behavior of the signals, which itself may cause errors while building the inequalities of the ROCRSSI. The errors arise from violation of the core idea of the ROCRSSI that is based on the assumption of homogeneity of radio transmission in a large range of direction. Thus, as stated by Liu et al. [1], the ROCRSSI does not accommodate radio irregularity and can make incorrect estimations and create wrong rings. However, the grid-scan algorithm aiming to define the CoG of the rings' intersection helps to reduce the influence of the wrong rings (the grid-scan algorithm is described in more details in Section 3.2.3). Consequently, as concluded by Liu et al. [1], the ROCRSSI generates small localization errors even if very irregular radio propagation takes place.

3.2.2 Refinements of the original ROCRSSI

The original ROCRSSI proposed by Liu et al. [1] has several drawbacks. During the following years after the proposition of the ROCRSSI these drawbacks have been specified. Thus the refinements of the original algorithm denoted as ROCRSSI+ [13] and ROCRSSI++ [10] have appeared. Description of the two named refinements and drawbacks that they overcome are given below.

The ROCRSSI+ proposed by Crepaldi et al. [13] is designed to handle the situations when a tag lies outside of the determined circles. The original ROCRSSI ignores such kind of situation, since it assumes that the area outside of the determined circles is unbounded (infinite) and no improvement to the tag's localization could be achieved in this case.



(a) ROCRSSI **(b)** ROCRSSI+

Figure 3.3: Difference between ROCRSSI and ROCRSSI+ while considering the area outside of the determined circles

Let us again assume a system consisting of tree fixed nodes A, B and C and a tag T, which location has to be found (Figure 3.3). Let us also assume that after measurement of RSSIs performed by anchors that are parts of the fixed nodes the following inequalities have been built: $RSSI_{CA} > RSSI_{BA} > RSSI_{TA}$, $RSSI_{AB} > RSSI_{CB} > RSSI_{TB}$ and $RSSI_{TC} > RSSI_{AC} > RSSI_{BC}$. The original ROCRSSI does not take into account the first two inequalities when the signal strengths received by the fixed nodes A and B from the tag T are lower than the signal strengths received by the named fixed nodes from the rest of two fixed nodes (B and C for the fixed node A, and A and C for the fixed node B). The first two inequalities mean that the tag T most probably lies outside of the circles with radiuses AB and BC and centers at the fixed nodes A and B respectively. The only one inequality, the original ROCRSSI takes into account, is the third one. According to this inequality the tag T most probably resides inside of the circle with center at the fixed node C and radius CA. As a result, the localization algorithm locates the tag T at the CoG of the intersection area formed by this circle only. Consequently, the estimated location E of the tag T coincides with the fixed node C (Figure 3.3a).

On the contrary, the refined localization technique ROCRSSI+ proposed by Crepaldi et al. [13] assumes that the area where the fixed nodes and the tag are deployed is bounded. This assumption allows to take into account the area outside of the determined circles. As a result, the ROCRSSI+ considers the first two inequalities, and the estimated location E of the tag T is calculated as CoG of the hatched intersection area showed in Figure 3.3b. The estimated location E in this case is affected by a smaller error. In addition, Crepaldi et al. [13] note that every time the tag resides outside of the determined circles, it also should be considered to be placed inside the bounded deployment area.

The further refinement of the original ROCRSSI localization technique has been proposed by Frattini et al. [10] and denoted by them as ROCRSSI++. The authors name inefficiencies that have still not been overcome by the previous refinement ROCRSSI+. They define the causes for each of the inefficiencies and name solutions for their overcoming:

1. *Inconsistency of the results, grounded on different order of the algorithm's inputs:* The inconsistency may take place in case of two channels with equal RSSI values that correspond to different distances. Under a channel the authors understand a communication link between two fixed nodes, which are separated by a known distance. In order to overcome such kind of inconsistency, the authors suggest to consider more reliable the RSSI value estimated on the longer distance. Since the RSSI values of the two channels are equal, the RSSI value estimated on the shorter distance is considered to be affected by more obstacles than the RSSI value estimated on the longer distance. The latter is considered to be more dependent only on distance and less on the met obstacles.

2. *Variable RSSI:* Unsteadiness of the RSSI values over the time is caused by reflection, refraction and interferences that affect wireless communications. The authors propose to compute RSSI as an average of several RSSI values measured during some period of time. If the RSSI values show a strong standard deviation, the authors also propose to apply a weighted mean by considering greater RSSI values having greater weights. The reason that the greater RSSI values have greater weights and, as a result, affect the average RSSI in a greater degree is the same as has been stated in the previous

paragraph: such kind of values are considered to be more dependent only on distance and less on the met obstacles.

3. *Channel asymmetry:* The channel asymmetry appears when, for example, the signal strength emitted by the fixed node A and received by the fixed node B ($RSSI_{AB}$) is not equal to the signal strength emitted by the fixed node B and received by the fixed node A ($RSSI_{BA}$). The ROCRSSI++ localization technique considers the channels between the fixed nodes to be symmetric. The greatest RSSI value is chosen as a reciprocal RSSI value of the channel.

4. *Memory and communication inefficiency:* The authors emphasize how to pack the matrix of RSSI values and the matrix of distances between each couple of fixed nodes that, according to the system's architecture of Frattini et al. [10], are stored by every asset beacon (tag). The main factor that reduces size of the stored matrices is the channels' symmetry. That is, in the channel formed by two fixed nodes A and B the following equality takes place: $RSSI_{AB} = RSSI_{BA}$. Therefore, the matrix of RSSI values diminishes to a triangular form. The same is true for the matrix of distances.

3.2.3 Grid-scan algorithm

The grid-scan algorithm is intended to define the CoG of the intersection area the rings form.

The original version of the grid-scan algorithm has been presented by He et al. [12]. The algorithm in general is able to define the CoG of any intersection area. In the following the algorithm is described as it has been implemented in this master's thesis.

With the grid-scan algorithm, as its name implies, the test venue is partitioned onto the imaginary cells, which themselves form a grid (Figure 3.4). For each of the anchors deployed within the bounded area the grid-scan algorithm goes through the formed grid and increments values of the cells which lie within the area (a circle, a ring or an area outside of the circle with the biggest radius the anchor is able to form), where according to the anchor the tag resides (tag's residential area). He et al. [12] have not mentioned in their paper, how to determine whether a cell lies within the tag's residential area or not. In this master's thesis it is considered that a cell belongs to the tag's residential area, if the cell's center of gravity (namely, the intersection of the cell's diagonals) lies within it.

The cells, which centers lie within the intersection of all of the tag's residential areas (within the intersection area), will have the greatest values (value 3 in Figure 3.4).

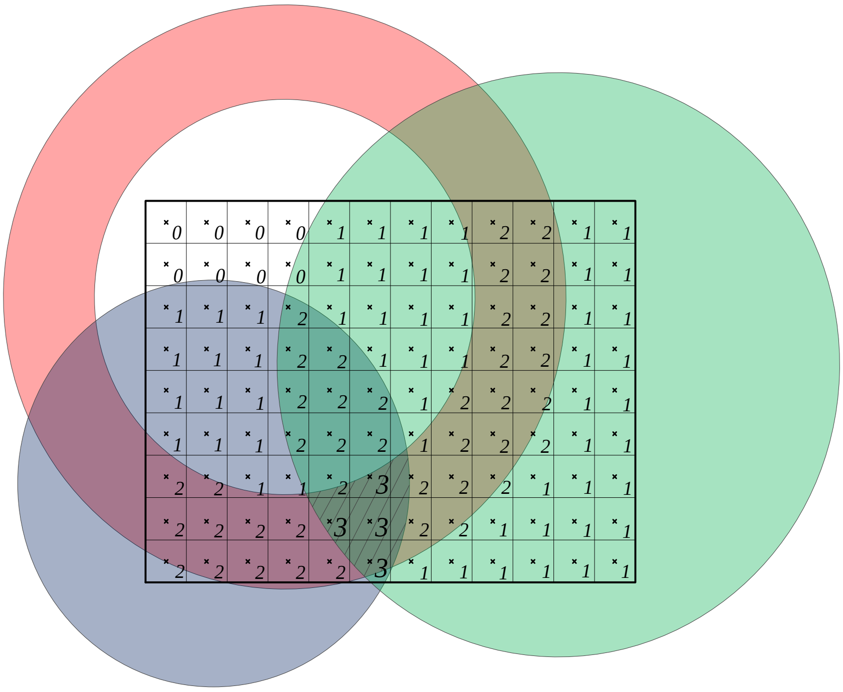


Figure 3.4: Core idea of the grid-scan algorithm defining the CoG of the intersection area

On the next step the grid-scan algorithm calculates the CoG of cells with the greatest values (the CoG of intersection area that is represented by these cells). Coordinates $x\,[m]$ and $y\,[m]$ of the CoG are calculated according to Equations 3.1:

(3.1a)

$$x = \frac{1}{n}\sum_{i=1}^{n} x_i$$

(3.1b)

$$y = \frac{1}{n}\sum_{i=1}^{n} y_i,$$

where n is the number cells that represent the intersection area, $x_i\,[m]$ and $y_i\,[m]$ are correspondingly x and y coordinates of center of the i^{th} cell from the set of cells that represent the intersection area.

The calculated CoG represents the estimated location of the tag.

Liu et al. [1] state in their research that the grid-scan algorithm helps to reduce influence of the wrong rings that could be generated by the ROCRSSI. Let us assume that more than a half

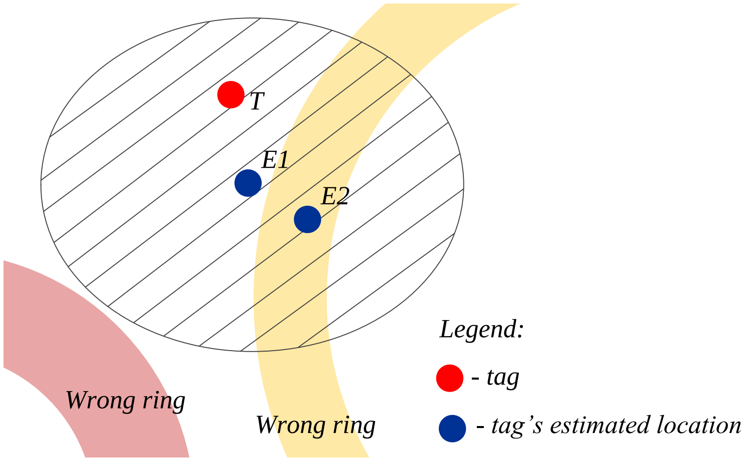


Figure 3.5: Grid-scan algorithm diminishes influence of wrong rings

of the generated rings are correct and form an intersection area shown as a hatched ellipse in Figure 3.5. The majority of the correct rings guarantees that values of the cells, which centers lie within the hatched area, are greater than values of other cells of the grid. In the case if even all of the wrong rings have no intersection with the hatched area and form their own (wrong) intersection area, the tag's estimated location (marked as E_1 in Figure 3.5) will be calculated as CoG of the hatched ellipse and the wrong rings will not be taken into account. In the case if some of the wrong rings intersect with the hatched area, the final intersection area may not contain the tag. However, the final intersection area lies within the hatched area, which is reasonably small if an ample number of anchors provide RSSI measurements as input for the localization algorithm. Therefore the CoG of the final intersection area (marked as E_2 in Figure 3.5) will lie next to the actual location of the tag T. Thus the grid-scan algorithm diminishes or even eliminates influence of the wrong rings.

Frattini et al. [10] have researched dependency of the localization accuracy of the system on granularity (size of cells) of the grid-scan algorithm. The localization accuracy increases with the cells' size reduction. However, the authors have discovered that the increase of the localization accuracy holds true only until certain dimensions of the cells (particularly until 41.5 cm in the environment where the authors have conducted their experiments). The further cells' size reduction has not shown noticeable localization improvements.

In order to increase the localization accuracy, Frattini et al. [10] also suggest to specify a threshold for the number of cells with maximal values. The cells with maximal values are considered as a result intersection if their number is greater than the specified threshold. However, the authors do not explain how an appropriate threshold has to be chosen and adjusted according to the current parameters of the system (e.g. granularity of the grid-scan algorithm, number of involved anchors etc.).

3.3 Localization strategies

In Section 2.4 two research concepts have been proposed. The first concept aims at leveraging automatic movement that is available in industry (e.g. a conveyor) for the purpose of simplification of the required calibration procedure. The second concept uses range-free localization technique with fixed nodes that are spatially distributed into emitters and receivers. Investigation of the concepts has the purpose to identify approaches for the ILS design that minimize (or even eliminate) the required calibration effort while achieving an accuracy that is able to reduce search times on the shop floor. Based on these two concepts research questions have been formulated in the same section. In order to cover both of the concepts and address the research questions the following localization strategies are proposed:

1. Localization for the case, when fixed nodes of the system are spatially merged (when every anchor is spatially merged with a reference node).
2. Localization for the case, when fixed nodes of the system are spatially merged. In addition preliminary calibration of the system assisted by an automatic movement that is available on industrial shop floors (e.g. by a conveyor) is proposed.
3. Localization for the case, when anchors and reference nodes are spatially distributed.
4. Localization for the case, when anchors and reference nodes are spatially distributed. In addition preliminary calibration of the system assisted by an automatic movement (e.g. by a conveyor) is proposed.
5. Localization utilizing anchors only. Preliminary calibration of the system with the help of an automatic movement (e.g. of a conveyor) is used in this case as well.

The listed strategies are based on the core idea of the ROCRSSI range-free localization technique described in Section 3.2. However, in the implemented ILS the core idea of the ROCRSSI has been applied with several modifications and extensions:

1. In the original ROCRSSI the localization algorithm is performed by a tag that operates with a matrix of measured RSSIs and a matrix of distances between the fixed nodes in order to determine possible area, where it is located. In the implemented ILS the localization load has been moved to the server. All measurements are performed by anchors. The measured RSSIs together with supplemental data that is necessary for the localization algorithm are sent by the anchors to the server.
2. The original ROCRSSI and its refinements consider an anchor (signals' receiver) and a reference node (signals' emitter) to be spatially merged. In the implemented ILS, besides the traditionally merged anchors and reference nodes, their spatially distributed deployment is investigated.

3. An automatic movement (line-following robot or conveyor) is leveraged in the system as a tool for the preliminary calibration of the system. With the help of the automatic movement, fingerprints (spatially anchored RSSIs) are collected at several points along the robot's motion path or along the conveyor. The original ROCRSSI and its refinements have not yet been used fingerprints for the localization purposes. In the implemented ILS this adaptation is investigated.

3.3.1 Localization strategy 1: spatially merged fixed nodes

The strategy totally coincides with the core idea of the ROCRSSI range-free localization technique described in Section 3.2 and is illustrated in Figure 3.6.

Let us assume that the fixed nodes FN_1, FN_2 and FN_3 illustrated in Figure 3.6 have formed the following inequalities after each of them had measured RSSI values of the tag T and of the respective two other fixed nodes: $RSSI_{FN_2FN_1} > RSSI_{TFN_1} > RSSI_{FN_3FN_1}$, $RSSI_{TFN_2} > RSSI_{FN_1FN_2} > RSSI_{FN_3FN_2}$, $RSSI_{TFN_3} > RSSI_{FN_1FN_3} > RSSI_{FN_2FN_3}$, where $RSSI_{MN}$ – signal strength received by the fixed node N and initially emitted by the fixed node (or tag) M. The inequalities imply that the tag T most probably is located inside the hatched area that is an intersection of the ring with center at the fixed node FN_1 and inner and outer radiuses FN_1FN_2 and FN_1FN_3 respectively and two circles with centers at FN_2 and FN_3 and radiuses FN_2FN_1 and FN_3FN_1 respectively. The tag's estimated location E is calculated as a CoG of the hatched area.

3.3.2 Localization strategy 2: spatially merged fixed nodes with automatic movement enabled calibration

Automatic movement (a conveyor) is used as a tool for collection of fingerprints. During the calibration procedure a tag is placed onto the powered conveyor, and then its RSSIs are measured by an anchor. Each time, when RSSI value is obtained from the tag, the anchor reads out its own timestamp. Having the read timestamp and knowing start time of the calibration procedure, conveyor's deployment (its coordinates within the test venue) and speed, one can easily calculate current coordinates of the moving tag. Thus fingerprints (spatially anchored RSSIs) are acquired by every anchor. The fingerprints are stored in the server's DB and used by the anchors for generation of additional rings in the localization algorithm later on. The fingerprints' acquisition procedure is described in detail in Section 3.4.

Figure 3.7 illustrates usage of fingerprints F_4 and F_7 for generation of an additional ring by the anchor that is a part of the fixed node FN_2. The additional rings are able to shrink the intersection area and thus, theoretically, to improve the system's localization accuracy. For instance, in Figure 3.7 two tag's estimated locations E_1 and E_2 are shown. E_1 has been calculated as a CoG of the intersection area derived without usage of the additional ring. E_2

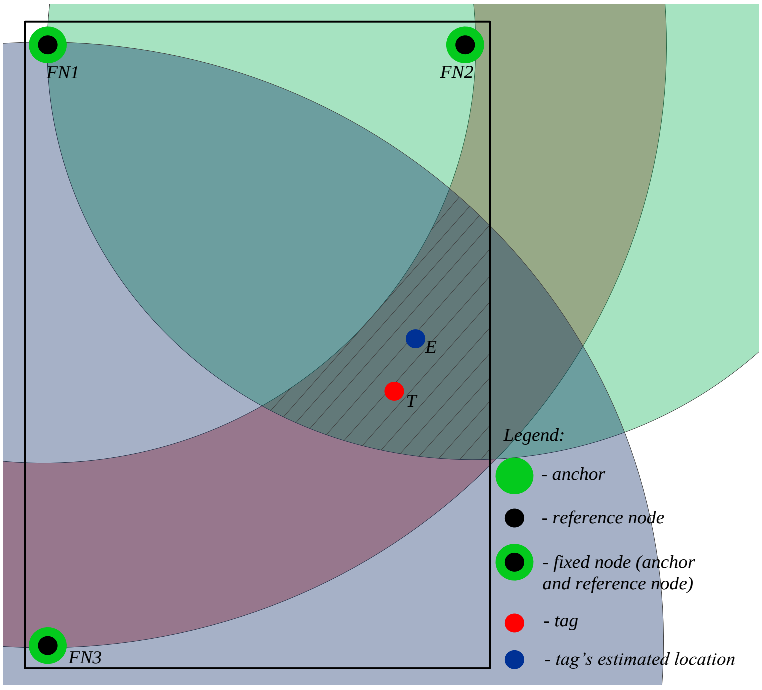


Figure 3.6: Example of localization strategy 1

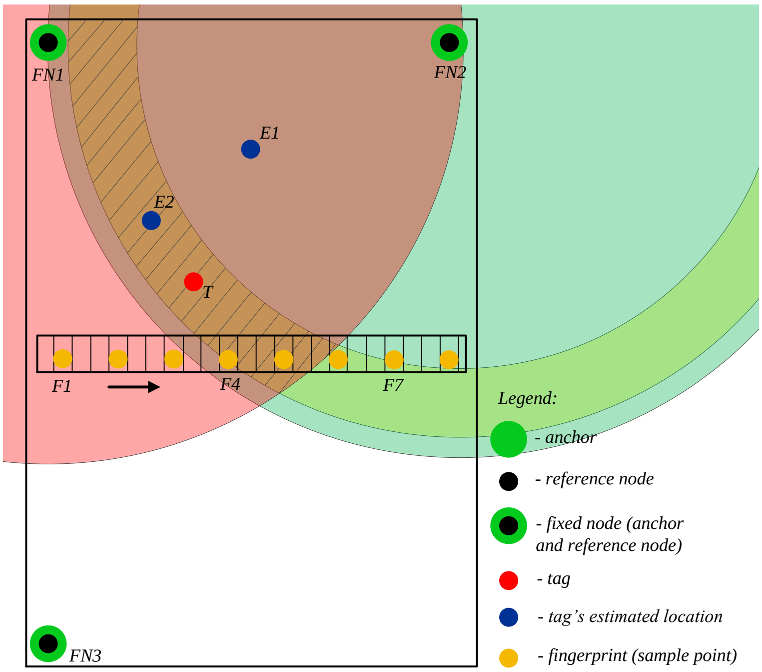


Figure 3.7: Example of localization strategy 2

has been calculated as a CoG of the intersection area that has been shrunk by the additional ring (hatched intersection area). As a result, the draft in Figure 3.7 shows that E_2 lies closer to the actual location of the tag T than E_1.

The strategy is illustrated in Figure 3.7.

3.3.3 Localization strategy 3: spatially distributed anchors and reference nodes

Spatial distribution of the fixed nodes into the signals' emitters (reference nodes) and signals' receivers (anchors) allow to increase variety of rings that could be generated by every anchor. For example, the anchor A_1 in Figure 3.8 is able to generate 4 variants of rings: a circle with radius A_1RN_1, a ring with inner and outer radiuses A_1RN_1 and A_1RN_2 respectively, a ring with inner and outer radiuses A_1RN_2 and A_1RN_3 respectively and an area that lies outside of the circle with radius A_1RN_3. For comparison, the anchor that is a part of the fixed node FN_1 in Figure 3.7 is able to generate only 3 variants of rings: a circle with radius FN_1FN_2, a ring with inner and outer radiuses FN_1FN_2 and FN_1FN_3 respectively, an area that lies outside of the circle with radius FN_1FN_3. The increase of variety of generated rings is able to shrink the intersection area and thus, theoretically, to improve the system's localization accuracy. Moreover, the reference nodes (standard BLE beacons) are cheap in comparison with the anchors (BLE compatible Android smartphones). That gives an opportunity to increase the number of deployed reference nodes as well, which increases variety of rings that could be generated by every anchor even more.

The strategy is illustrated in Figure 3.8.

3.3.4 Localization strategy 4: spatially distributed anchors and reference nodes with automatic movement enabled calibration

This strategy combines the concept of leveraging automatic movement that is typically available on industrial shop floors (in particular, a conveyor) for the calibration purposes with the concept of spatial distribution of the fixed nodes into emitters and receivers. The concepts have been involved into the localization strategies presented in Sections 3.3.2 and 3.3.3 respectively.

The strategy is illustrated in Figure 3.9.

3.3.5 Localization strategy 5: automatic movement enabled calibration without reference nodes

Theoretically, only the fingerprints acquired during the calibration procedure could be used for the rings' generation. As a result, presence of anchors only is sufficient. The strategy is of

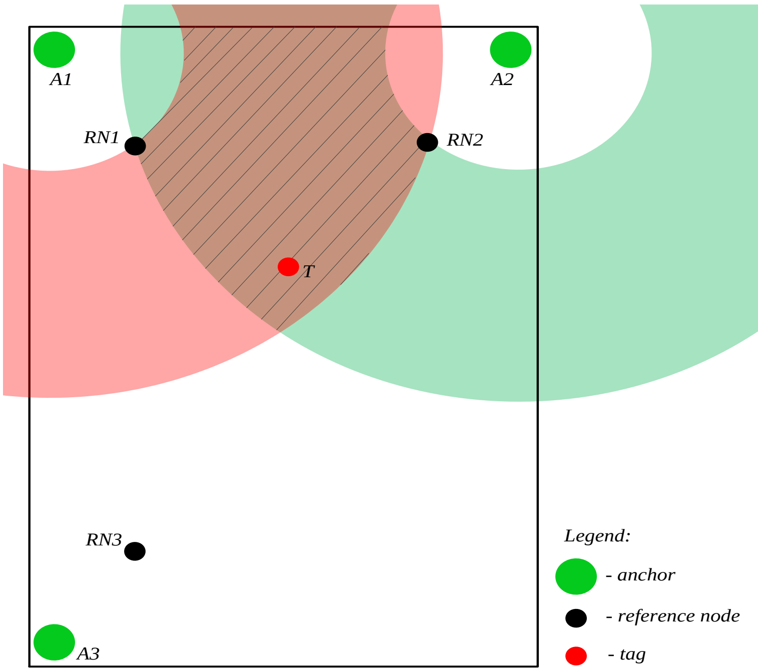


Figure 3.8: Example of localization strategy 3

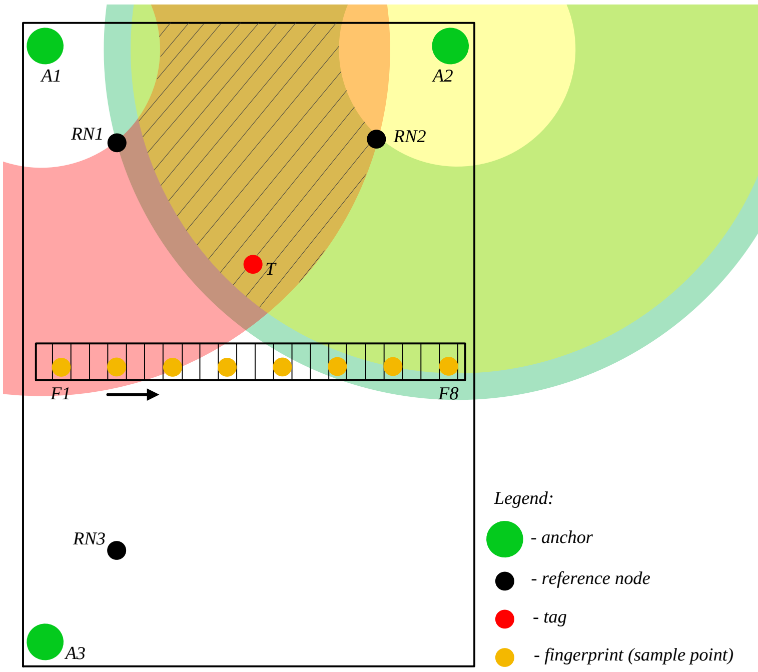


Figure 3.9: Example of localization strategy 4

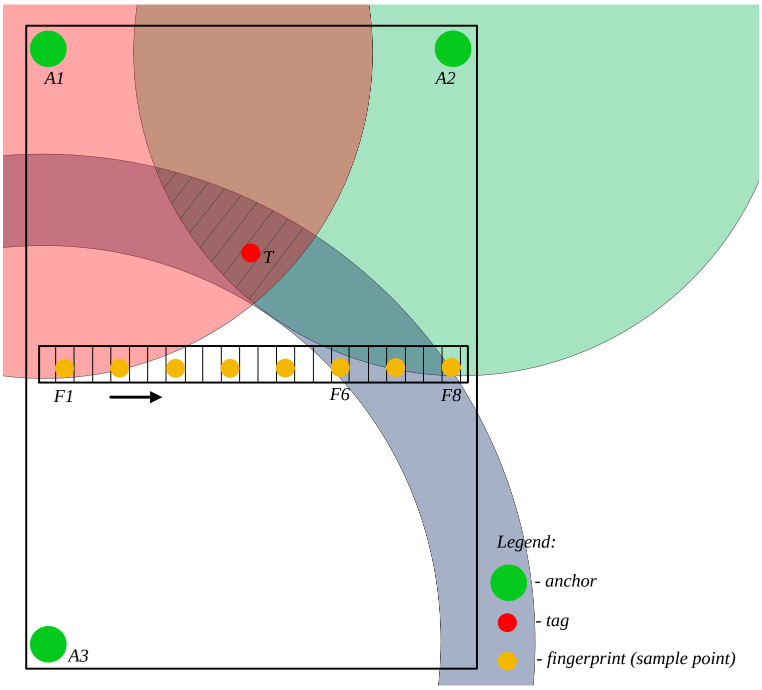


Figure 3.10: Example of localization strategy 5

value for a comprehensive investigation of the research concepts formulated in Section 2.4. Since localization will be based on the preliminary acquired fingerprints and will not use real time RSSI measurements for the rings' generation, the strategy could help to evaluate influence of changes within the test venue (moving humans, changes in the equipment positions etc.) on the localization accuracy of the system.

The strategy is illustrated in Figure 3.10.

3.3.6 Hypothesis

Localization strategies 2, 3 and 4 that have been described in Sections 3.3.2, 3.3.3 and 3.3.4 correspondingly and that leverage research concepts named in Section 2.4 (namely, usage of automatic movements for the purpose of preliminary calibration of the system and spatial distribution of fixed nodes into the signals' receivers (anchors) and signals' emitters (reference nodes)) offer common hypothesis that is stated in the following paragraph.

Increase in number of reference points of the system (at the expense of obtained fingerprints or spatial distribution of fixed nodes that can be supplemented with extra reference nodes) promotes increase in variety of rings and circles that can be constructed by anchors of the system. As a result, the rings' and circles' intersection area, which CoG is considered as an estimated location of the sought tag, should be shrunk. Thus, theoretically, localization accuracy of the system is improved.

However, it should be also noted here that, since underlying ROCRSSI localization technique operates with geometrical intersection of rings and circles, localization accuracy of the system should theoretically depend not only on the number of reference points, but also on the deployment of anchors, reference nodes and motion path of the source of automatic movement (e.g. conveyor). Dependency of localization accuracy of the system on both of the factors that have been named above constitutes separate research topic and is not investigated as a part of this thesis.

3.4 Fingerprints' acquisition algorithm

Some of the localization strategies proposed and described in Section 3.3 (namely, localization strategies 2, 4 and 5) consider an automatic movement (in particular, a conveyor) as a tool for preliminary calibration of the system. The calibration procedure is performed by periodically scanning for the tag that is placed onto the automatically moved device (e.g. work piece carrier on a conveyor) and obtaining its RSSIs. Every time when RSSI value is obtained from the tag, the scanning device reads out its own timestamp. Having the read timestamp and knowing start time of the calibration procedure as well as the motion path and the speed of the automatic movement, one can easily calculate current coordinates of the moving tag. Thus fingerprints (spatially anchored RSSIs) are acquired. The fingerprints are stored in the DB and used as additional reference points of the system later on (see descriptions of localization algorithm and localization strategies provided in Sections 3.2 and 3.3 correspondingly).

Let us consider a conveyor as a tool for the preliminary calibration of the system. The mathematical apparatus that is used in the fingerprints' acquisition procedure and that is described below with reference to a conveyor can be easily used for any other source of automatic movement (e.g. for a line-following robot), which motion path consists of straight, constant-speed sections connected in series.

As a visual example, a conveyor that consists of three sections is shown in Figure 3.11. A tag is placed onto the powered conveyor and moves along it. Let us consider the N^{th} section of the conveyor. The length L_N $[m]$ of the section can be calculated according to Equation (3.2):

$$L_N = \sqrt{(x_{NE} - x_{NB})^2 + (y_{NE} - y_{NB})^2}, \quad (3.2)$$

where x_{NB} $[m]$ and y_{NB} $[m]$ are correspondingly x and y coordinates of the section's beginning, x_{NE} $[m]$ and y_{NE} $[m]$ are correspondingly x and y coordinates of the section's end.

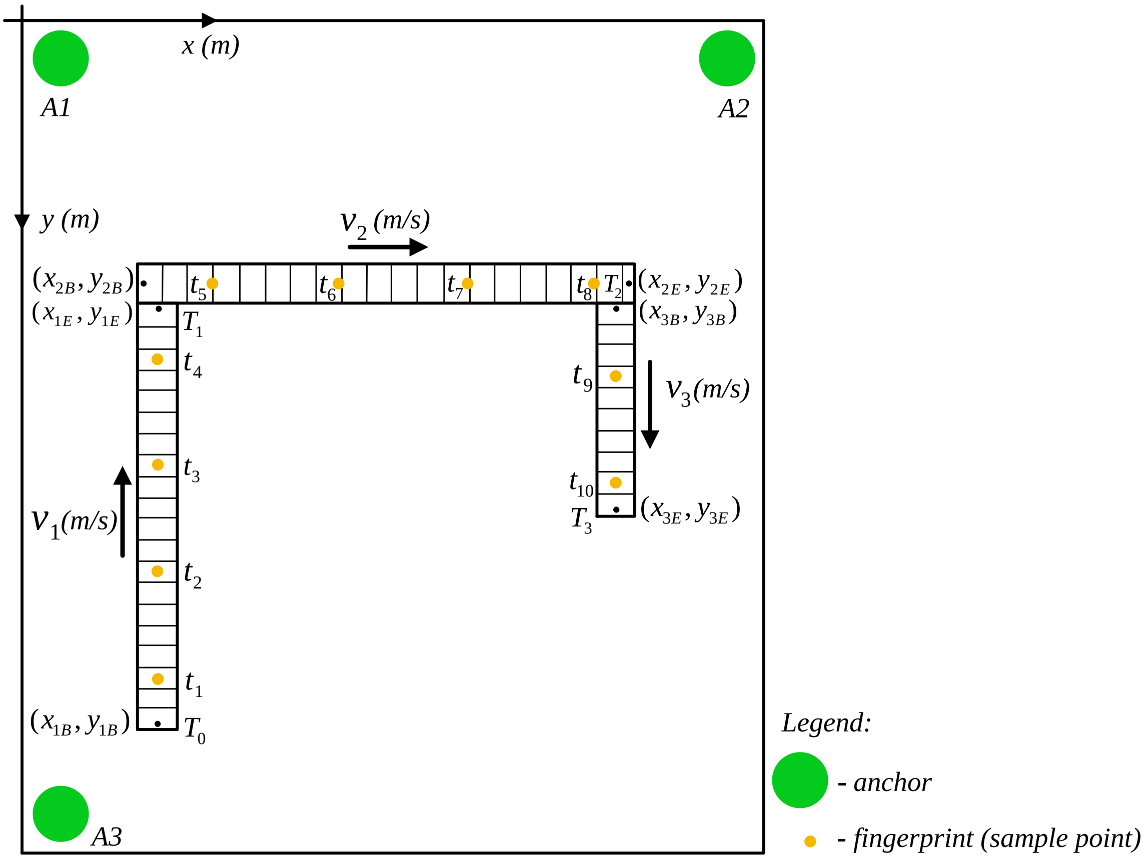


Figure 3.11: Example of fingerprints' acquisition algorithm

Timestamp T_N $[ms]$ of the moment when tag should theoretically reach the end of the N^{th} section can be calculated according to Equation (3.3):

$$T_N = T_0 + \sum_{i=1}^{N} \frac{L_i}{v_i} \cdot 1000, \tag{3.3}$$

where T_0 $[ms]$ is timestamp of the moment, when the calibration procedure starts (when the tag is placed in the beginning of the 1^{st} section), L_i $[m]$ is length and v_i $[m/s]$ is speed of the i^{th} section. Such a theoretical timestamp can be calculated for every section of the conveyor and used as a time boundary that signals to the algorithm that the section is theoretically passed by the tag.

While the tag is moving along the conveyor, its RSSIs are obtained. Let us consider that when the m^{th} RSSI value is obtained from the tag, the scanning device reads out its own timestamp t_m $[ms]$. Let us also consider that $t_m \in (T_{N-1}, T_N]$. That in fact means that at this moment of time the tag is theoretically moving along the N^{th} section of the conveyor. Then current x and y coordinates of the tag (namely, $x_m\,[m]$ and $y_m\,[m]$) can be calculated according to Equations 3.4:

(3.4a)

$$x_m = x_{NB} + \frac{v_N(t_m - T_{N-1})}{1000} \cdot \frac{x_{NE} - x_{NB}}{L_N}$$

(3.4b)

$$y_m = y_{NB} + \frac{v_N(t_m - T_{N-1})}{1000} \cdot \frac{y_{NE} - y_{NB}}{L_N},$$

where $x_{NB}\,[m]$ and $y_{NB}\,[m]$ are correspondingly x and y coordinates of the beginning of the section that is theoretically being passed at the current moment of time (N^{th} section of the conveyor), $x_{NE}\,[m]$ and $y_{NE}\,[m]$ are x and y coordinates of its end, $L_N\,[m]$ and $v_N\,[m/s]$ are correspondingly length and speed of the section, $t_m\,[ms]$ is timestamp of the moment, when m^{th} RSSI value has been obtained from the tag, $T_{N-1}\,[ms]$ is timestamp of the moment, when the tag has passed the end of the previous conveyor's section.

When the tag's coordinates are calculated, the obtained RSSI value is considered to be anchored to some specific point of the test venue. In other words, a fingerprint is considered to be formed. Such a fingerprint together with some supplemental data is sent to the server's DB later on. The acquired fingerprints play role of additional reference points of the system. From the implementation point of view the fingerprints' acquisition procedure is described in more details in Section 4.2.3. Its GUI is presented in Section 4.2.1.

3.5 Noise filtering

Akeila et al. [3] state in their paper about sensitivity of RF signals in general and BLE signals in particular to disturbances due to reflections, shadowing and fading. Sensitivity of the RF signals leads to significant fluctuations of their measured RSSIs. When the RSSIs are used for the purpose of localization, the fluctuations cause localization errors.

Radius Networks highlight another factor for localization errors: orientation of both the smart phone and the BLE device [15]. They state also that the biggest factor affecting accuracy of localization is radio noise. Strong RF signals that have high signal-to-noise ratio (SNR) are less affected by the noise than the weak ones that have low SNR. For this reason, the closer is a BLE device, the more stable RSSIs can be obtained from it and as a result the more accurate localization can be performed.

Radius Networks further propose a simple filtering for noise. They recommend to use a running average of RSSIs, e.g. always to operate with measurements that have been received during the most recent 20 seconds. From these measurements particularly high and low numbers are common to be filtered out. The algorithm that is used in the Android Beacon

Library[2] created by Radius Networks rejects 10% of the highest and lowest values measured over the 20 second period and calculates average of the rest. Radius Networks note that their filtering approach helps to achieve more accurate and stable localization results if scanning Android and emitting BLE devices are immovable. However, they also state about the drawback of such kind of filter: since a set of previously obtained RSSI values is involved into the calculations, the system reacts on any movements of scanner or/and emitter with a delay [15].

Every anchor operating in the implemented ILS performs a scan for BLE devices. The main idea is to get a list of currently available BLE devices together with RSSIs obtained from them. The first sessions of experiments have confirmed sensitivity of BLE signals to any changes of environment. High variance of obtained RSSIs has led to a high variance of localization accuracy. Hence filtering was introduced to solve this issue.

The filter has been built into the scan procedure that is performed by every anchor. As well as the filter proposed by Radius Networks, the implemented one works with sets of RSSIs. The implemented filter has borrowed two core ideas from its Radius Networks' counterpart: it rejects 10% of particularly high and low RSSI values of the set and averages the rest of the values. The idea of running average of RSSIs proposed by Radius Networks has been replaced by the idea of their discrete accumulation during a preset time interval. For the discrete accumulation of RSSIs the implemented scan procedure provides for several consecutive scan rounds. During every scan round RSSIs of available BLE devices are obtained and accumulated. The accumulated RSSIs are filtered only after the predefined number of scan rounds. Implementation details of the realized filter built into the scan procedure are described in Section 4.2.2.

Since both of the filters (the Radius Networks' filter and the implemented one) work with sets of RSSI values (the first one operates with continuously running average of RSSI values, the second one – with discretely accumulated sets of RSSIs), they deprive the system of its ability to react to any movements of scanning Android or/and emitting BLE device(s) on-the-fly. For this reason the filter has been disabled in the case of fingerprints' acquisition that is performed with the help of an automatic movement. The fingerprints' acquisition algorithm is described in Sections 3.4 and 4.2.3.

3.6 Optimized beacon configuration

Localization accuracy of the system can be improved not only by a filtering for noise that is considered in Section 3.5, but also by a proper configuration of beacons. Radius Networks recommend to configure beacons as follows [15]:

[2] `https://altbeacon.github.io/android-beacon-library/configure.html`

1. Set up maximal output power level supported by the beacon, since stronger signals have higher SNRs and are less affected by the noise.
2. Use the highest transmission frequency supported by the beacon. Radius Networks give the following reason for this: higher transmission frequency allows a scanning device to obtain more RSSIs that will be sorted, filtered and averaged later on in order to filter out noise.

TI CC2541 SensorTags that are used in the implemented ILS as BLE signals' emitters provide programmable output power range from -23 dBm to 0 dBm [16]. For all of the TI CC2541 SensorTags acting in the system as reference nodes and as a sought tag maximal output power of 0 dBm has been set.

TI CC2541 SensorTag has programmable RF range starting from 2379 MHz and ending on 2496 MHz. Transmission frequency of the TI CC2541 SensorTag can be programmed in 1 MHz steps [16]. However, this can be done with digressions from the base BLE specification. Gupta [17] describes in his book that BLE uses 40 RF channels that start from 2402 MHz and are spaced 2 MHz apart. Moreover, for advertising only three RF channels (namely, channels 0, 12 and 39 that are placed at frequencies 2402 MHz, 2426 MHz and 2480 MHz correspondingly) are provided. The advertising channels are spread far apart in order decrease probability of complete interference from other devices: if one of the advertising channels is interfered, then at least the other two are available for advertising. Therefore, in order to avoid interference it is better to leave advertising scheme as it is provided by the BLE specification and do not intentionally set maximal transmission frequency.

4 System implementation

This chapter explains, how the system design proposed in the previous chapter has been implemented. The structure of the chapter is organized as follows: Section 4.1 explains conventions of naming of interfaces, classes and methods that appear in the following description of the system implementation; Section 4.2 provides description of the implemented client's Android application (its GUI and logic of its main procedures that are, namely, the scan and the fingerprints' acquisition ones); in Section 4.3 structure of the server's DB and implementation logic of the grid-scan algorithm performed by the server are described; the last Section 4.4 covers the question of the client-server communication.

4.1 Naming conventions

In the following description of the ILS implementation it is necessary to distinguish interfaces, methods and classes provided by Android and Java application programming interfaces (APIs) from the implemented ones. Therefore these naming conventions are introduced:

1. Any interface or method that is provided by Android or Java API will be mentioned in the following description of the system's implementation together with its package prefix. E.g. public final class `BluetoothAdapter` from the package `android.bluetooth` will be mentioned in the text as `android.bluetooth.BluetoothAdapter` or public interface `Runnable` from the package `java.lang` will be referred as `java.lang.Runnable`.
2. All implemented interfaces and methods are placed into a single package, therefore the package's name can be omitted. The implemented interfaces and methods will be mentioned in the text without their package prefix. E.g. public class `NodesDeployment` or public interface `AsyncResponseForMainActivity`.
3. Any method (either provided by API or implemented one) will be mentioned together with the class it belongs to. E.g. public method `startScan(List<ScanFilter>filters, ScanSettings settings, ScanCallback callback)` of the public final class `android.bluetooth.le.BluetoothLeScanner`.
4. If a method is an implementation of the interface's method or if a method is overridden then this will be mentioned individually. E.g. overridden public method `onScanResult(int callbackType, ScanResult result)` of the public abstract class `android.bluetooth.le.ScanCallback`.

4.2 Client side

Every client of the implemented ILS is represented by a Motorola Moto G 2014 smartphone controlling by Android 5.0 Lollipop and with the implemented application onboard. The implemented Android application can be subdivided into two functional parts. The first one is intended for filling the server's DB by configuration data. The configuration data is subsequently used by the localization algorithm. The second one partakes in the localization procedure: it periodically scans for the available BLE devices (reference nodes and the tag), forms a scan result that is subsequently sent to the server's DB, reads from the server new estimated location of the tag that has been calculated based on the scan results received from the system's clients. It also provides a graphical representation of the currently conducted experiment.

This section, first of all, presents the GUI of the implemented Android application. Then the logic of the scan procedure is described in detail. Finally, the logic of the fingerprints' acquisition procedure that obtains fingerprints (spatially anchored RSSIs) from the tag moving along a defined motion path of an automatic movement is explained.

4.2.1 Overview of GUI

4.2.1.1 Experiment setup

Setting up experiments aims at accumulating the configuration data that is subsequently used by the localization algorithm. The data is: parameters of the test venue, where experiments are conducted; data related to the deployment of anchors and reference nodes and to the deployment of conveyor, if an automatic movement (e.g. a conveyor) is used for the purpose of preliminary calibration of the system; data related to the fingerprints that have been acquired with the help of the automatic movement.

Setting up experiments is organized as a step-by-step process. Every step includes setup of necessary parameters and sending them to the server's DB. Each of the steps is implemented on a separate screen. Navigation between the screens is organized by means of swiping.

The initial step involves configuration of the test venue's parameters: its name, size and degree of granulation. The test venue's name represents a string identifier that is used for representation of every test venue in the dropdown lists of test venues later on. The size parameter implies the test venue's width and length in meters. The degree of granulation represents a number of cells along the test venue's width and length. The granulation onto the cells is necessary for the further performance of the grid-scan algorithm that is a part of localization algorithm. Sending of the configured parameters to the server that handles their writing to the DB is triggered by pressing the "Create" button. GUI of the test venue's configuration step is depicted in Figure 4.1.

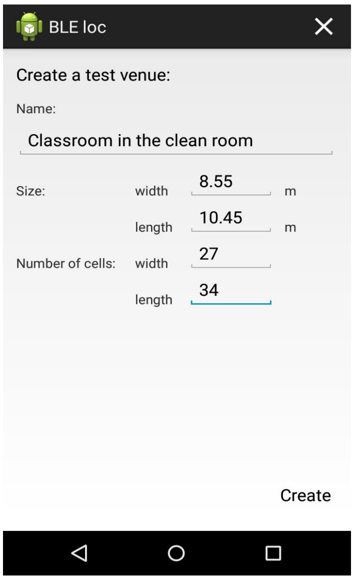
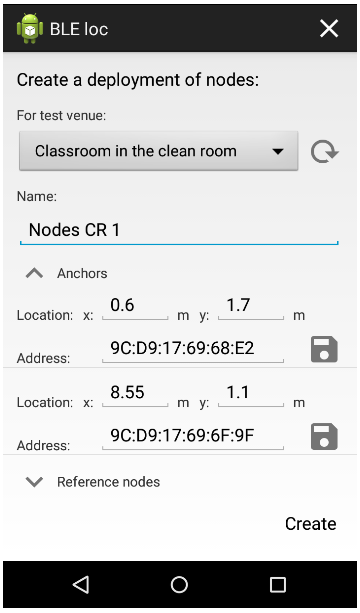
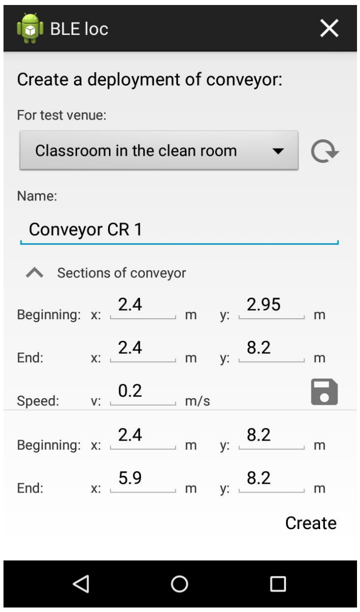


Figure 4.1: Android client application GUI – configuration of test venue



(a) Deployment of nodes



(b) Deployment of conveyor

Figure 4.2: Android client application GUI – configuration of nodes' and conveyor's deployments

After a test venue is configured, nodes (anchors and reference nodes) and conveyor can be deployed within it.

The GUI of the nodes' deployment configuration is depicted in Figure 4.2a. The nodes' deployment configuration includes selection of the test venue, where the nodes are going to be deployed, from a dropdown list of already configured and saved test venues, setup of the deployment's name and configuration of lists of anchors and reference nodes. The lists of anchors and reference nodes can be expanded by clicking on the chevrons to the left of the corresponding lists' titles. The following parameters are able to be set for every anchor and reference node: x- and y-coordinates in meters, Bluetooth address. The configured deployment of nodes is sent to the server by clicking the "Create" button. The server handles writing of the deployment to the DB.

The GUI of the conveyor's deployment configuration is presented in Figure 4.2b. The conveyor's deployment configuration involves the same logic as the previously described configuration of the nodes' deployment: a conveyor is deployed within an already configured and saved test venue, and a name of the conveyor's deployment is set. The conveyor is considered to consist of straight, constant-speed sections connected in series. It is necessary to mention here that instead of conveyor any other source of automatic movement (e.g. a robot) can be used. The idea of subdivision of the total motion path of the robot into straight, constant-speed path sections connected in series stays unchanged in this case. In the application conveyor sections are represented as an expandable list. The following configuration parameters are provided for every section of the conveyor: x- and y-coordinates of the section's beginning and end in meters, the section's speed in meters per second. The configured deployment of conveyor is sent to the server that handles its writing to the DB by clicking the "Create" button.

With the help of the conveyor (or any other source of automatic movement) a collection of fingerprints is formed. A collection of fingerprints is a container intended for fingerprints that will be subsequently acquired by the clients of the system. The GUI that is meant for the creation of the fingerprints' collection is shown in Figure 4.3. Every collection of fingerprints is related to a previously configured and saved deployment of a conveyor. Analog to the previous configuration steps, the configuration of the fingerprints' collection demands setup of its name (the string identifier that is used for representation of every fingerprints' collection in the dropdown list of fingerprints' collections later on). Sending of the configured fingerprints' collection to the server is triggered by pressing the "Create" button. The server handles writing of the collection to the DB.

On the basis of the saved configuration data experiments are created. Every experiment represents a test venue with deployed nodes (anchors and reference nodes), conveyor (if applicable) and with collection of fingerprints formed with the help of the conveyor. The GUI of the experiment's configuration is depicted in Figure 4.4. Configuration parameters of the experiment (namely, a test venue, one of its nodes' and one of its conveyors' deployments, a fingerprints' collection related to the chosen conveyor's deployment) are set with the help of

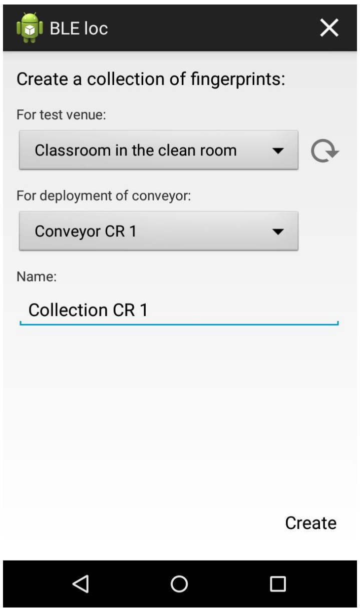


Figure 4.3: Android client application GUI – configuration of fingerprints' collection

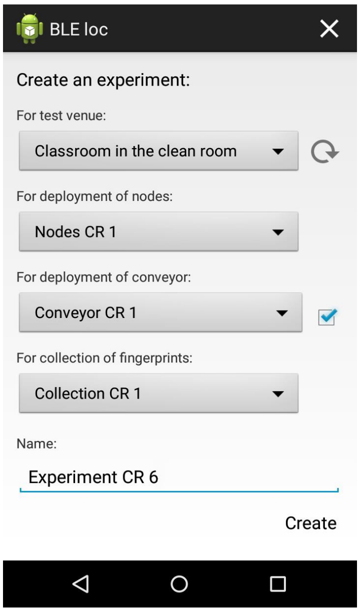


Figure 4.4: Android client application GUI – configuration of experiment

the dropdown lists, where the names of the previously saved configurations are presented. The GUI provides an opportunity to switch on/off usage of conveyor for preliminary calibration of the system using a check box. For instance, for the localization strategies 1 and 3 that are described in Sections 3.3.1 and 3.3.3 correspondingly the conveyor has to be switched off. The localization strategies 2, 4 and 5 that are explained in Sections 3.3.2, 3.3.4 and 3.3.5 respectively on the contrary demand a conveyor enabled calibration. For the experiment a name has to be specified. The configured experiment is sent to the server that handles its writing to the DB by clicking the "Create" button. The preliminary configured and saved experiments are able to be selected from a dropdown list later on, when the scan for the available BLE devices is prepared to be performed by the client.

4.2.1.2 Fingerprints' acquisition

The GUI of the fingerprints' acquisition procedure is presented in Figure 4.5. The fingerprints are acquired for an in advance created collection of fingerprints. Every anchor of the system operates with its own fingerprints later on. Therefore, the fingerprints' acquisition procedure has to be performed by every anchor individually. The procedure is started/stopped by clicking the "Start"/"Stop" buttons correspondingly. At the same time with the start of the fingerprints' acquisition procedure, the tag is placed in the beginning of the first section of the powered conveyor (or, in case of usage of a line-following robot, the powered robot is placed in the beginning of its motion path). Since the lengths and the speeds of the conveyor's sections are known to the system, the implemented ILS is able to calculate the time that the tag needs for passing the conveyor. As the time passes, the application notifies a user about the necessity to stop the fingerprints' acquisition procedure by means of a toast. The theoretical basis of the fingerprints' acquisition procedure is described in Section 3.4. The logic of its implementation is provided in Section 4.2.3.

4.2.1.3 BLE device scan

The scan for the available BLE devices starts from selection of an experiment from a dropdown list of preliminary configured and saved experiments. Based on the data of the selected experiment a current one is created. The current experiment is created for every new examined actual location of the tag. Thus, besides the data of the selected experiment the current one includes also coordinates of the actual location of the tag and the flag that indicates whether the actual location takes part in the further calculations (the case, when localization errors are calculated) or not (the case, when localization is performed without calculation of the localization errors). The GUI that is meant for the selection of the experiment and for the creation of a current experiment is depicted in Figure 4.6a. The configuration data of the selected experiment (namely, parameters of the test venue, data related to the nodes' and conveyor's deployments, data related to the collection of fingerprints) is displayed to a user

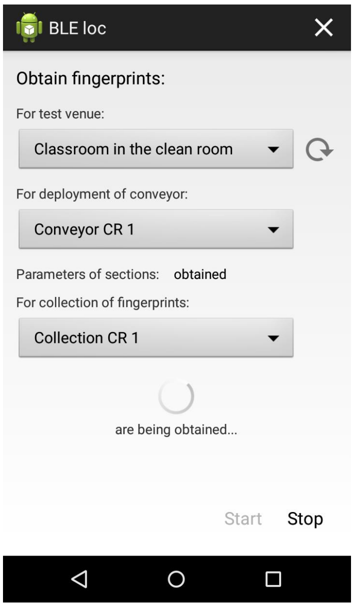
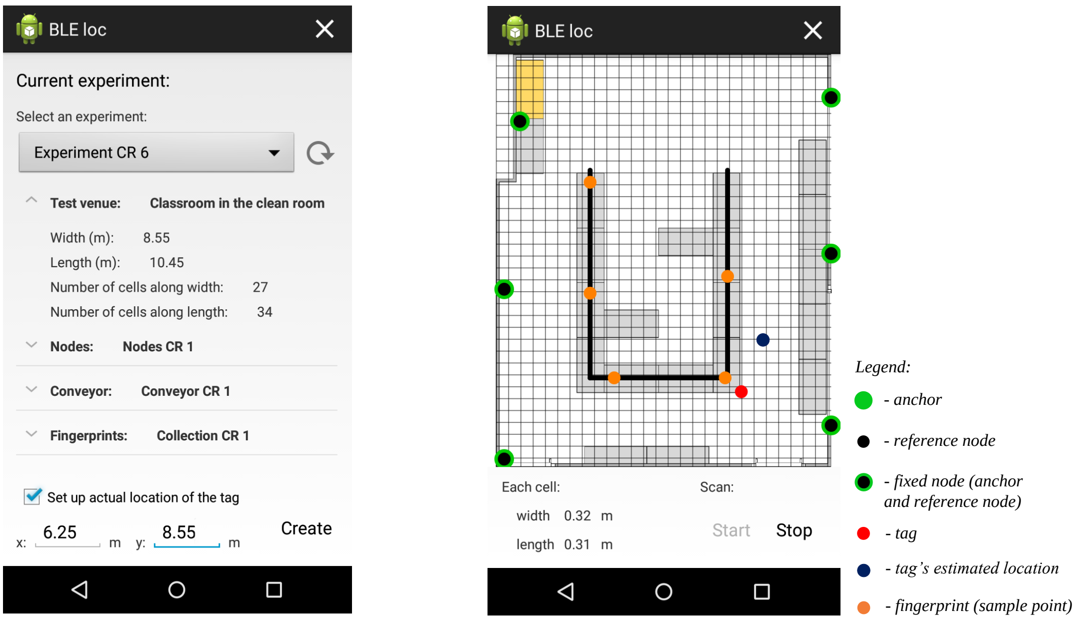


Figure 4.5: Android client application GUI – acquisition of fingerprints



(a) Configuration of current experiment **(b)** Graphical representation of current experiment

Figure 4.6: Android client application GUI – configuration of current experiment and its graphical representation

in the form of expandable lists. Usage of actual location of the tag in the further calculations is switched on/off with the help of a check box. Such kind of option allows to use the application both for research (then usage of the actual location of the tag is switched on and localization errors are calculated) and for demonstration purposes (then usage of the actual location of the tag is switched off and only localization without calculation of its errors is performed). Configuration of the current experiment is sent to the server by clicking the "Create" button.

As soon as the current experiment is created, the scan for the available BLE devices can be started. The GUI that is intended for the control of the scan procedure as well as for graphical representation of the currently conducted experiment is shown in Figure 4.6b. The scan procedure can be started/stopped by pressing the "Start"/"Stop" buttons correspondingly. The graphical representation reflects the test venue, where the current experiment is conducted, as well as deployments of nodes (anchors and reference nodes) and conveyor with fingerprints acquired by this particular client along the conveyor's path (if applicable). It also displays both actual location of the tag and its estimated location that is renewed periodically in the course of the scan procedure. The graphically represented test venue is granulated onto the cells. This cell granulation is necessary for performance of the grid-scan algorithm that is a part of the localization algorithm. The grid-scan algorithm is described in Sections 3.2.3 and 4.3.2. As a background, a schematic sketch of the test venue is displayed to a user. The graphical symbols that are used by the application correspond to the symbols considered for representation of components of the system throughout this master's thesis.

4.2.2 Scan procedure logic

4.2.2.1 Differences in the scan procedures of Android 4.4 KitKat and Android 5.0 Lollipop

During implementation of the Android application for indoor localization Android 5.0 Lollipop for Motorola Moto G 2014 smart phone has been introduced. Android 5.0 provides new capabilities for usage of the BLE technology that has been initially introduced in Android 4.3 Jelly Bean. Devices that operate with Android 4.3 and the later Android 4.4 support BLE technology in the central role. That means that Android devices can scan for the BLE signals, but not broadcast them. Starting from Android 5.0 an Android device is also able to act as a peripheral device, namely to broadcast the BLE signals [18].

The new peripheral role of the Android devices allows to extend the functionality of the anchors. Being managed by Android 5.0, anchors are also able to act as additional reference nodes, augmenting the number of reference points of the ILS and thus, theoretically, increasing its localization accuracy. This additional functionality has not been implemented in the frame of this master's thesis, however it is listed in Section 6.2 as a suggestion for further extension of the system.

BLE APIs that are provided by Android 5.0 differ from those that have been used in the previous versions of Android (Android 4.3 and Android 4.4). Omitting the new peripheral functionality of Android 5.0, let us consider its intrinsic properties while acting in the central role.

In Android 5.0 `startLeScan(BluetoothAdapter.LeScanCallback callback)` and `stopLeScan(BluetoothAdapter.LeScanCallback callback)` methods of the `android.bluetooth.BluetoothAdapter` class have been deprecated and replaced with a new API. The new API provides instruments for more control over the scanning procedure. In Android 5.0 the scanning procedure can be configured with the help of the `android.bluetooth.le.ScanFilter` class that allows to adjust an application to scan for only the specific types of devices it is interested in. Another parameter that is used for the scanner's configuration belongs to the `android.bluetooth.le.ScanSettings` class, which allows to set a scan mode [19].

Android 5.0 provides the following scan modes [20]:

1. `SCAN_MODE_LOW_POWER`: performs BLE scan in low power mode. This mode is used in Android 5.0 by default.
2. `SCAN_MODE_LOW_LATENCY`: provides the highest duty cycle for the BLE scan. Android documentation recommends to only use this scan mode when the application is operating in the foreground [21].
3. `SCAN_MODE_BALANCED`: performs BLE scan in balanced power mode. Return of the scan results is done with a rate that provides a balance between scan frequency and power consumption.

The application that has been created in this master's thesis operates in the foreground. Moreover, the ability to deliver scan results with low latencies can be considered as an improvement for the application. As a result the `SCAN_MODE_LOW_LATENCY` has been leveraged.

For starting the scan procedure the list of scan filters, scan settings and implemented `android.bluetooth.le.ScanCallback` that reports whenever a new BLE `android.bluetooth.le.ScanResult` is available are passed into the `startScan(List<ScanFilter>filters, ScanSettings settings, ScanCallback callback)` method of the `android.bluetooth.le.BluetoothLeScanner`. For stopping the scan procedure the `stopScan(ScanCallback callback)` method of the `android.bluetooth.le.BluetoothLeScanner` is called.

4.2.2.2 Scan procedure core idea

Execution flows of the scan procedure and of the subsequent tag's estimated location reading are described in Section 4.4.4.[1] Below scan procedure is reviewed from the perspective of its logic.

For the scan procedure the `android.app.Service ServiceOperationWithDevices` is responsible. The service is started/stopped from the `MainActivity` by clicking the "Start"/"Stop" buttons correspondingly (see GUI depicted in Figure 4.6b). Depending on the received command, the service is either instantiated and started or destroyed. An activity diagram that presents control flows of the `ServiceOperationWithDevices` is depicted in Figure 4.7.

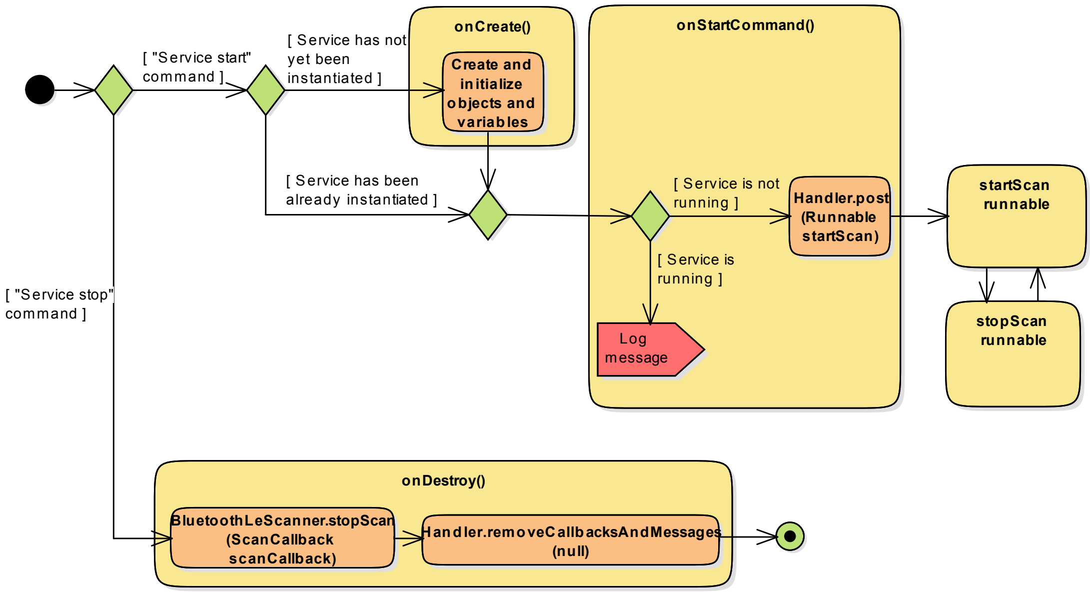


Figure 4.7: Activity diagram illustrating the scan procedure

The service is started from the addition of the `java.lang.Runnable startScan` to the message queue of the current thread with the help of the created `android.os.Handler` that is attached to the thread. Another runnable that is executed in the frame of the `ServiceOperationWithDevices` is the

[1]Sequence diagram presenting execution flows of the scan procedure and of the subsequent tag's estimated location reading is depicted in Figure C.4 in appendices.

stopScan runnable. Each of the two mentioned runnables adds another one to the thread's message queue with a delay. Thus, a cyclic execution of the runnables is achieved.

The destruction of the service is accompanied by the call of the `stopScan(ScanCallback scanCallback)` method of the `android.bluetooth.le.BluetoothLeScanner` and by removing of all of the runnables and messages from the thread's message queue (`android.os.Handler.removeCallbacksAndMessages(null)`).

The logic of the `startScan` and the `stopScan` runnables is presented and briefly described below.

The activity diagram that presents control flows of the `startScan` runnable is depicted in Figure 4.8. The `startScan` runnable starts from increase of the scan counter that stores the number of already performed scan rounds. Several scan rounds are run continuously in order to accumulate RSSI values for their further filtering. Then the BLE scan is started by calling the `startScan(List<ScanFilter>scanFilters, ScanSettings scanSettings, ScanCallback scanCallback)` method of the `android.bluetooth.le.BluetoothLeScanner`. The scan results of the available BLE devices are delivered through the created `scanCallback`. In the end the `stopScan` runnable is added with a delay to the thread's message queue.

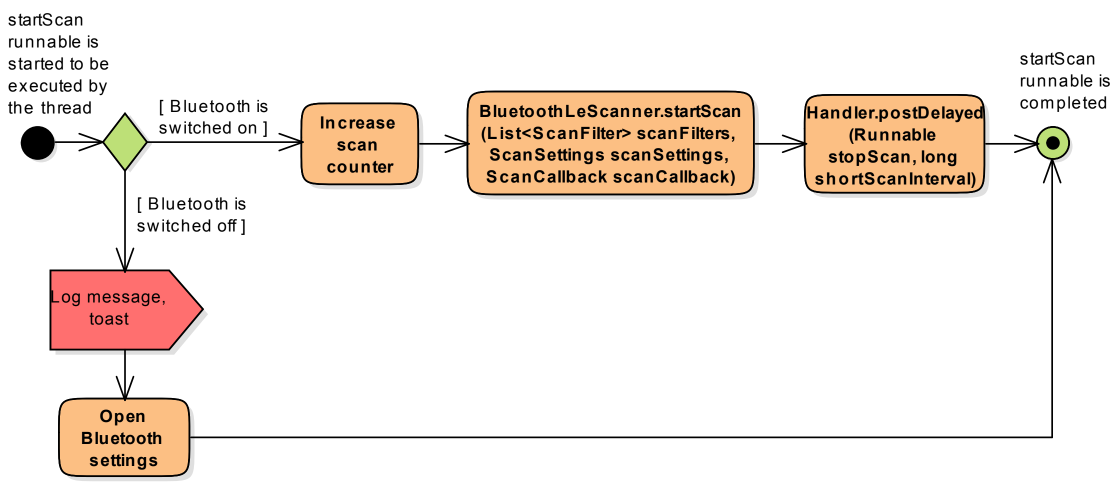


Figure 4.8: Activity diagram: `startScan` runnable

The `stopScan` runnable is intended to process the reference nodes' and the tag's scan results that have been accumulated during several rounds of the BLE scan procedure. Its activity diagram is shown in Figure 4.9. After stopping the BLE scan, the `stopScan` runnable checks whether all of the specified scan rounds have been passed or not. If the predefined number of scan rounds has not yet been reached, then the `startScan` runnable is added to the thread's message queue and started to be executed. Otherwise, the `stopScan` runnable starts to operate with the accumulated scan results. At first, a filter is applied to the accumulated scan results of every

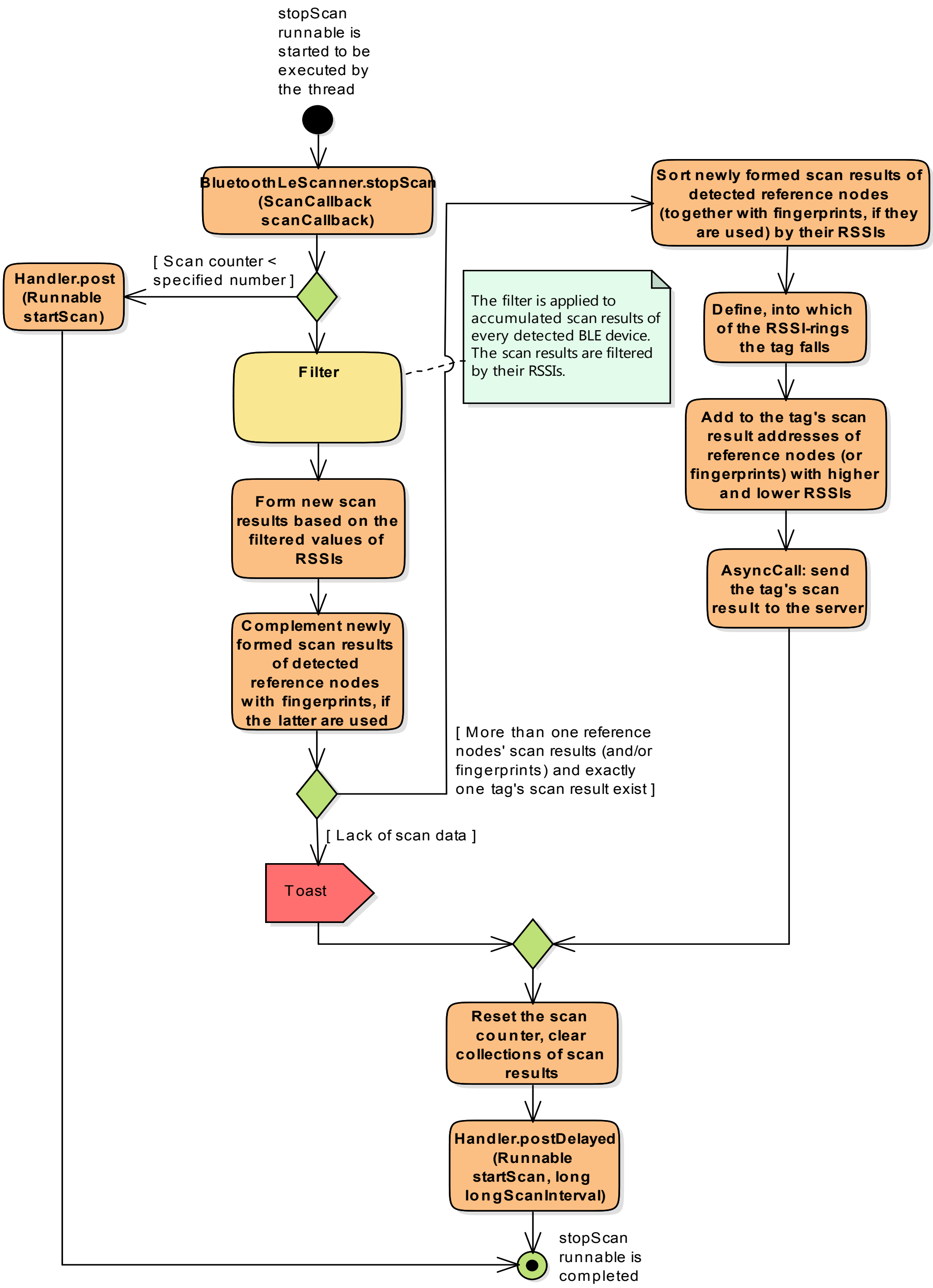


Figure 4.9: Activity diagram: `stopScan` runnable

detected BLE device. The scan results are filtered by their RSSIs. Based on the filtered values of RSSIs the new scan results are formed. The newly formed scan results of reference nodes are complemented with fingerprints (if the latter are used) and sorted by their RSSIs. Then the core idea of the ROCRSSI localization technique that is described in detail in Section 3.2.1 is performed: an RSSI inequality that shows between which of the reference nodes and/or fingerprints the tag theoretically locates is built. In the end, the tag's scan result is enriched with addresses of reference nodes (or fingerprints) with higher and lower RSSIs and sent to the server. After operation with the scan results, the `stopScan` runnable adds with a delay the `startScan` runnable to the thread's message queue. In such a way a loop of the runnables' execution is organized.

It has been found empirically that a short latency of the scan rounds together with a large number of their repetitions provides good compromise between the total amount of time that is spent on the scan and the number of accumulated scan results. As a result, the following latencies and the number of scan rounds have been considered in the implemented ILS: the latency of a scan round has been set to 200 milliseconds, the number of scan rounds that have to be performed in order to accumulate data for its further filtering has been set to 100, the time delay between the sets of scan rounds has been set to 7 seconds. However, even such a good compromise has allowed to accumulate only 10–15 scan results for their further filtering after 100 of 200 milliseconds scan rounds taking in total 20 seconds. The desirable behavior of accumulating a large amount of scan results during a short period of time was impossible to achieve because of the properties of the BLE scan in Motorola Moto G 2014 Android smartphones that serve as clients of the system.

4.2.2.3 Scan callback

Every time, when a new BLE device is detected, the `android.bluetooth.le.ScanCallback` reports. During a scan round that lasts 200 milliseconds every BLE device is detected at most once. The overridden `onScanResult()` method of the implemented scan callback delivers an `android.bluetooth.le.ScanResult`. The delivered scan result provides data of the detected BLE device (its name and address) as well as an RSSI from it. Every delivered scan result is filtered by its name. Since all of the BLE beacons operating in the system (the reference nodes and the tag) are represented by the TI CC2541 SensorTags, only these devices are admitted for their further operation. In the following step, the scan callback reads the client's current time (represented as a timestamp) and forms a new scan result based on the read timestamp as well as on the address of the detected BLE device and on the RSSI from it.

The formed scan result is put into the linked hash map of scan results. As keys of the linked hash map, addresses of the detected BLE devices are used. Values of the linked hash map are represented by lists of formed scan results. During the specified 100 scan rounds the value-lists of the linked hash are filled out by formed scan results of corresponding BLE devices. Thus, accumulation of scan results for their further filtering is performed. The linked

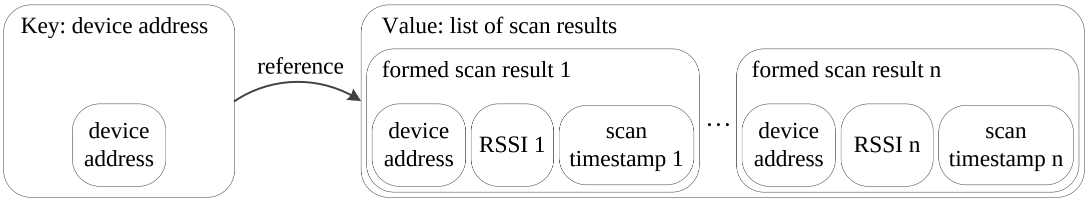


Figure 4.10: Key-value mapping of the linked hash map of scan results

hash map of scan results has been implemented in order to allow the filter to easily accesses accumulated scan results of every detected BLE device by address of the device. A key-value mapping of the linked hash map of scan results is presented in Figure 4.10.

4.2.2.4 Filter

After the predefined number of scan rounds are passed, a filter is applied to RSSIs of the scan results that have been accumulated during these scan rounds (Figure 4.9). The filter is applied to RSSIs of every detected BLE device. The filter has been implemented in order to reduce variance of obtained RSSIs that led to a variance of the system's localization accuracy. From the theoretical point of view, the question of filtering is viewed in Section 3.5. An activity diagram illustrating the implemented filter is depicted in Figure 4.11. The filtering implies the following steps: first of all, accumulated scan results of every detected BLE device are sorted by their RSSIs; then 10% of scan results with particularly high and low RSSIs are filtered out; in the end, arithmetic average of RSSIs of the rest of scan results is calculated. The arithmetic average is considered as a more accurate value reflecting RSSI between the considered BLE device (reference node or tag) and an anchor and is used in further logic of the `ServiceOperationWithDevices`.

Figure 4.11: Activity diagram illustrating the filtering of accumulated scan results

4.2.3 Fingerprints' acquisition procedure logic

The fingerprints are acquired with the help of an automatic movement (e.g of a conveyor or a line-following robot) and used as additional reference points of the system by the further executed localization algorithm. During the fingerprints' acquisition procedure a tag is placed onto the powered conveyor or onto the moving line-following robot. For the fingerprints' acquisition the `android.app.Service` `ServiceFingerprintsAcquisition` is responsible. The core idea of the service's implementation corresponds to the core idea of the scan procedure described in Section 4.2.2. To be exact, the service is organized as a cyclic execution of two `java.lang.Runnable` objects: `startScan` and `stopScan`. The service is started/stopped from the `MainActivity` by clicking the "Start"/"Stop" buttons correspondingly (see GUI depicted in Figure 4.5). Depending on the received command, the service is either instantiated and started or destroyed. The logic of the fingerprints' acquisition procedure differs from the logic of the scan procedure in the following:

1. Since fingerprints are acquired from a moving tag, the filter proposed and described in Section 3.5 can not be applied in this case. This also means that scan results obtained from the moving tag should not be accumulated before their further processing. The scan result that is obtained during execution of the `startScan` runnable is processed in the `stopScan` runnable later on. The latency of 7 seconds is set for execution of every runnable.

2. Processing of the scan result implies calculation of coordinates of the moving tag at the moment of the BLE signal's emission. Namely, a fingerprint (a spatially anchored RSSI) is formed. The mathematical apparatus of the coordinates' calculation is described in detail in Section 3.4. Briefly, the coordinates of the moving tag are calculated based on the following known parameters: coordinates of the conveyor's sections (or the robot's path sections) within the test venue, speed of the conveyor (or the robot), timestamps of the beginning of the fingerprints' acquisition procedure and of a point in time, when the scan result has been obtained from the moving tag. Each of the formed fingerprints is sent to the server and associated with data of the fingerprints' collection selected in the corresponding dropdown list (see GUI of the fingerprints' acquisition procedure depicted in Figure 4.5).

It should be mentioned here that the logic of the fingerprints' acquisition procedure has been implemented for a single conveyor (or a single motion path of a line-following robot) that represents straight, constant-speed sections (or path sections in the case of the robot's usage) connected in series. If the conveyor's sections (or the path sections of the robot's motion trajectory) are disconnected, then several conveyors (or several robot's motion trajectories) are considered in the system. The more general case considering usage of several conveyors (or several robot's motion trajectories) is listed in Section 6.2 as a suggestion for further extension of the system.

Execution flows of the client-server communication during the fingerprints' acquisition procedure are briefly described in Section 4.4.5.[2]

4.3 Server side

The server of the implemented ILS has been allocated on the Lenovo ThinkPad T430 laptop equipped with a Wi-Fi adapter for the server's connection to the established local wireless network. The server is responsible for persisting the clients' scan results, performing the localization algorithm, persisting the location estimates for their later analysis and transferring the location estimates to the clients (by their requests). The server allocates web services that are provided to the clients of the system also participating in the local wireless network and include necessary operations with the DB. The server has been implemented with the help of the Apache Tomcat v7.0. For implementation of the DB MySQL has been used.

This section introduces the structure of the server's DB and describes the grid-scan algorithm, which represents a core of the localization algorithm's logic.

4.3.1 DB structure

All entities (tables) of the DB are in fact represented by Java classes of the server's application and related to each other with the help of Hibernate annotations. The overall structure of the DB is subordinated to the logic that is described below.[3]

All instances of the test venues, where localization experiments are conducted, are represented in the DB by the `TerrainData` entity. Configuration of an experiment implies deployment of nodes (anchors and reference nodes) and of a conveyor (if a conveyor or any other source of automatic movement is used for the preliminary calibration of the system) within the desired test venue. Instances of the nodes' deployments are represented in the DB by the `NodesDeployment` entity, instances of the conveyor's deployments – by the `ConveyorDeployment` entity. Several nodes' and conveyor's deployments are able to be configured for the same test venue. Therefore, both of the entities have many-to-one relationship to the `TerrainData` entity.

As the names of the deployments imply, each of them represents a set of nodes (anchors or reference nodes) or conveyor's sections placed within the test venue. Instances of the deployed nodes and conveyor's sections are represented in the DB by the `Node` and the `ConveyorSection`

[2]Sequence diagram representing execution flows of the client-server communication during the fingerprints' acquisition procedure is depicted in Figure C.5 in appendices.

[3]DMD of the server's DB is depicted in Figure B.1 in appendices. DMD itself represents an extension of the class diagram.

entities correspondingly. Both of them have many-to-one relationship to their parent entities (`NodesDeployment` and `ConveyorDeployment` entities respectively).

With the help of a conveyor (or any other source of automatic movement) fingerprints (the spatially anchored RSSIs) are acquired from a tag placed onto it. In order to pool the fingerprints that are being acquired by the anchors of the system a collection of fingerprints is provided. With the help of the same conveyor the fingerprints are able to be acquired for different collections of fingerprints. All instances of the fingerprints' collections are represented in the DB by the `FingerprintsCollection` entity (many-to-one relationship to the `ConveyorDeployment`). Instances of fingerprints are represented by the `Fingerprint` entity (many-to-one relationship to the `FingerprintsCollection`).

It is necessary to mention here that every fingerprint has a dual nature. On the one hand, it can be considered as a node of the system with its calculated coordinates and an artificially formed address. On the other hand, it possesses an RSSI anchored to the coordinates. Thus, a fingerprint is also able to be considered as a scan result with the obtained RSSI, known scanner's addresses, time of obtaining etc. The latter is considered by clients. The former – by the server. Therefore, on the server side the `Node` and the `Fingerprint` Java classes extend the abstract `AbstractNode` class that provides coordinates and address as its variables. Each of the `Node` and the `Fingerprint` DB entities has one-to-one relationship to the `AbstractNode` entity.

All of the entities described above and related to each other provide data for configuration of experiments. Every experiment represents a test venue with deployed nodes (anchors and reference nodes), conveyor or any other source of automatic movement (if applicable) and with collection of fingerprints formed with its help. Instances of experiments are represented in the DB by the `Experiment` entity that has many-to-one relationships to both of the `NodesDeployment` and the `FingerprintsCollection` entities.

During the localization procedure a tag can be localized at several points within the test venue. The points, where the tag is placed for its further localization are called actual locations of the tag. Whenever the tag is placed at its new actual location, a new current experiment is considered to be started. All of the instances of current experiments are represented in the DB by the `CurrentExperiment` entity. Since within a test venue with deployed nodes, conveyor (if applicable) and preliminary acquired fingerprints several actual locations of the tag can be examined, the `CurrentExperiment` entity has many-to-one relationship to the `Experiment` one.

Localization (or calculation of estimated location of the tag) operates with the tag's scan results. Implementation logic of the scan procedure as well as formation of the tag's scan results is described in Section 4.2.2. Instances of the tag's scan results and of the estimated locations of the tag that are calculated with the help of the scan results received from the clients of the system are represented in the DB by the `ScanResult` and the `TagEstimatedLocation` entities correspondingly. Since during the localization procedure several scan results of the tag placed at its actual location are sequentially obtained, formed and sent to the server and, after the receiving of every scan result by the server, a new estimated location of the tag is calculated, both of the `ScanResult` and the `TagEstimatedLocation` entities have many-to-one relationships to the `CurrentExperiment` one.

As it has been already mentioned above, every estimated location of the tag is calculated by the server with the help of the scan results received from the clients of the system. In order to have an opportunity to trace, which of the received scan results have been used for calculation of the particular estimated location of the tag, an one-to-many relationship has been established between the `TagEstimatedLocation` and the `ScanResult` entities. The one-to-many relationship has been implemented with the help of the `estimated_location_scan_results` join table.

4.3.2 Grid-scan algorithm logic

The grid-scan algorithm determines intersection area of the rings and circles formed by the clients of the system and calculates its CoG (coordinates of estimated location of the tag). Theoretical basis of the grid-scan algorithm is provided in Section 3.2.3.

For the grid-scan's logic the `GridScan` class is responsible. An activity diagram of the implemented grid-scan algorithm is depicted in Figure 4.12. The grid-scan algorithm is initiated every time, when the server receives a scan result from a client. As an input parameter for the grid-scan algorithm, the currently conducted experiment is sent (the last instance of the `CurrentExperiment` DB entity). Every current experiment contains data of the configured experiment (parameters of the test venue, its nodes' and conveyor's deployments, data of fingerprints' acquired during the preliminary calibration of the system (if applicable)) as well as coordinates of the actual location of the tag and the `actualLocationEnabled` flag that indicates, in which of the two provided modes (namely, either in the research mode *with* or in the demonstration mode *without* calculation of localization errors) the experiment is currently conducted.[4]

In the research mode (`actualLocationEnabled == true`) the number of estimations that have to be performed as part of the current experiment (for the current actual location of the tag) is limited (see theory provided in Section 5.2). In the demonstration mode (`actualLocationEnabled == false`) the grid-scan's performance is not limited by the predefined number of estimations. The modes are able to be switched with the help of the "Set up actual location of the tag" check box on one of the screens of the implemented client's Android application (see GUI presented in Figure 4.6a).

Data of the formed rings is stored in the scan results sent by the clients to the server. Every scan result possesses `scannerAddress`, `higherRssiAddress` and `lowerRssiAddress` variables that contain addresses of nodes (anchors and reference nodes) and fingerprints serving as a center of

[4]See DMD of the server's DB provided in Figure B.1 in appendices.

the ring and as points on the ring's inner and outer circumferences correspondingly.[5] If a client locates the tag within a circle with the smallest radius, the `higherRssiAddress` is set to `no`. If the tag is located within an area outside of the circle with the biggest radius, then the `lowerRssiAddress` is set to `no`. In order to have an opportunity to operate with the received scan results (to map addresses provided by them on the corresponding nodes and fingerprints), the grid-scan algorithm reads currently operating nodes (anchors and reference nodes) as well as fingerprints acquired by the anchors (if applicable) from the DB and forms their linked hash maps using unique BLE addresses of anchors and reference nodes and artificially formed addresses of fingerprints as keys.

Scan results that are going to be further operated by the grid-scan algorithm are also read from the DB: for every anchor the last scan result obtained by the anchor in the current experiment is read. Operations with each of the read scan results include the following:

1. Every scan result is added to the collection of scan results of the tag's estimated location that is currently being calculated.[6]
2. From the earlier formed linked hash maps of anchors, reference nodes and fingerprints required anchor, reference node(s) and/or fingerprint(s) are obtained with the help of addresses provided by every particular scan result.
3. Coordinates of the obtained anchor, reference node(s) and/or fingerprint(s) are used for determination of the test venue's area (a circle with the smallest radius, a ring or an area outside of the circle with the biggest radius), where, according to the scan result, the tag locates.
4. Values of the test venue's cells, which centers belong to the determined area, are incremented.

After consideration of all of the read anchors' scan results, test venue's cells with maximal values are searched. These cells represent an intersection area of the rings and circles formed by the clients of the system. The CoG of the intersection area is considered as an estimated location of the tag.

If the system operates in the research mode (`actualLocationEnabled == true`), then the grid-scan algorithm calculates localization errors. Then the `TagEstimatedLocation` object is formed. The formed object is not written to the DB, if and only if in the research mode (`actualLocationEnabled == true`) the current number of performed estimations is greater *than* or equal *to* its predefined maximum. Otherwise, `maxNumOfEstimationsReached` variable of the formed `TagEstimatedLocation` object is additionally initialized (to `true`, if the system operates in the research mode and the formed instance of the `TagEstimatedLocation` class is the last permissible, and to `false`, otherwise).

[5]See the `ScanResult` entity depicted in Figure B.1 in appendices.

[6]See relationship between the `TagEstimatedLocation` and the `ScanResult` entities depicted in Figure B.1 in appendices.

Then the finally formed `TagEstimatedLocation` object is written to the DB with incrementation of the performed estimations' number. After formation of a scan result and sending it to the server, every client reads the most recent instance of the `TagEstimatedLocation` entity from the server's DB.[7] If the `maxNumOfEstimationsReached` flag of the read instance is set to `true`, then the client indicates a user about the necessity to stop the scan procedure.

[7]See sequence diagram depicted in Figure C.4 in appendices.

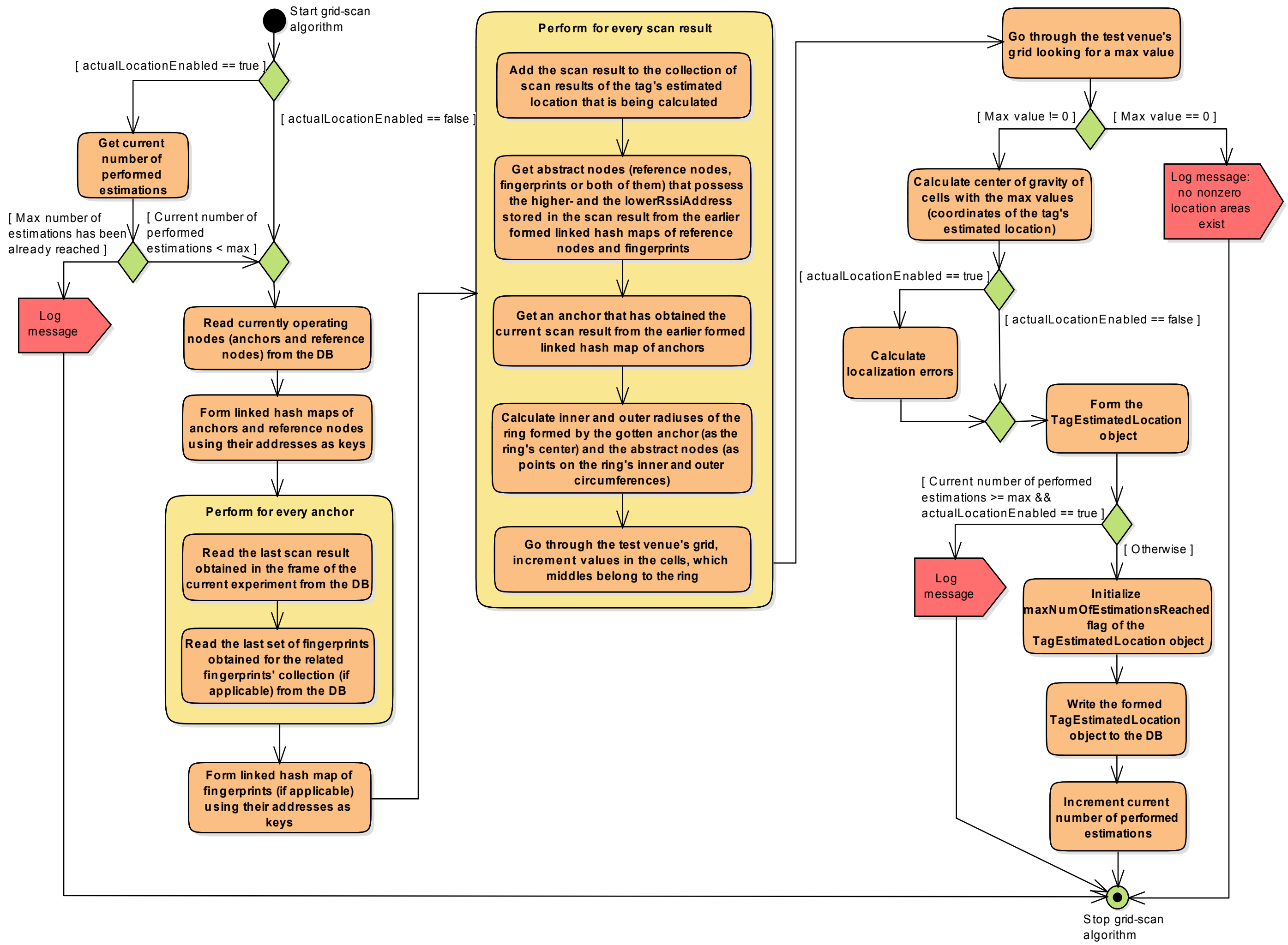


Figure 4.12: Activity diagram illustrating the grid-scan's implementation

4.4 Client-server communication

This section introduces the reader to the core logic and mechanisms as well as to the main scenarios and procedures of the client-server communication.

4.4.1 Core mechanisms of the client-server communication

Communication between the clients and the server of the system has been implemented with the help of web services. In the implemented ILS web services have been realized with the help of the Simple Object Access Protocol (SOAP).

Whenever a web service is needed to be invoked by a client, an instance of implemented asynchronous task's class extending `android.os.AsyncTask` is created and started to be executed. The web service is invoked from the executing instance. The `android.os.AsyncTask` class itself represents a convenient mechanism of performing background operations and publishing results on the user interface (UI) thread without having to manipulate threads and/or handlers [22]. Asynchronous tasks are created either in the `MainActivity` or in the `ServiceOperationWithDevices` and the `ServiceFingerprintsAcquisition` client's classes that need to communicate with the server via web services. Logic related to the web services' invocation is concentrated in methods of the client's `WebServiceClient` class. Web services themselves are implemented by methods of the server's `CommunicationWithDatabase` class and represent operations with the server's DB tables. Description of different scenarios of the client-server communication is provided below.[8]

4.4.2 Loading GUI data requests

One of the cases of the loading GUI data represents the "Read related data from DB" scenario. The general case of this scenario takes place, when a client fills out a dropdown list of configurations, which set depends on the configuration that has been previously selected in a dropdown list above. For instance, when user selects a test venue from a corresponding dropdown list, the system may require to read data of the conveyors' deployments that have been created *for* (are related *to*) the selected test venue (see, for instance, GUI representing configuration of a fingerprints' collection provided in Figure 4.3).[9] Other examples representing general case of the "Read related data from DB" scenario are: reading data of nodes' deployments related to a selected test venue (see GUI representing configuration of an experiment provided in Figure 4.4) and reading data of fingerprints' collections related to a selected

[8] Sequence diagrams representing different scenarios and cases of the client-server communication are depicted in Figures C.2, C.3, C.4 and C.5. Overview of components that are used in the sequence diagrams is presented in Figure C.1 in appendices.

[9] Sequence diagram illustrating load of GUI data by the example of reading data of conveyors' deployments that relate to the test venue with provided identifier (ID) is presented in Figure C.2 in appendices.

conveyor's deployment (see, for example, GUI representing preliminary configuration of the fingerprints' acquisition procedure provided in Figure 4.5).

A special case of the "Read related data from DB" scenario takes place, when a set of nodes, a set of conveyor's sections or a set of fingerprints that belong correspondingly to a particular nodes' deployment, conveyor's deployment or a fingerprints' collection is read from the server. The obtained sets are not presented in corresponding dropdown lists. However, they are stored on the client and used in further logic: by the fingerprints' acquisition or by the scan procedure. The client indicates a user about obtaining of any mentioned set by the word "obtained" on the corresponding screen of its Android application (see GUI provided in Figures 4.5 and 5.7). The core logic of the "Read related data from DB" scenario stays unchanged for both the general and the special cases.

The load of GUI data by the example of reading data of conveyors' deployments that relate to the test venue with provided ID is going to be briefly described below.[10] This scenario takes place, when, for instance, a fingerprints' collection is configured by a user (see GUI provided in Figure 4.3).

Whenever the user selects a test venue from the corresponding dropdown list (spinner) in order to configure a fingerprints' collection, the `onItemSelected()` method of the `MainActivity` class initiates reading data of the related conveyors' deployments. For this purpose, an instance of the implemented `AsyncCallReadConveyorsDeploymentsDataWS` class extending `android.os.AsyncTask` is created and started to be executed. As an input parameter, an object comprising ID of the spinner that has initiated the reading process and ID of the selected test venue is passed to the asynchronous task. The ID of the spinner is required in order to let the system know, which of the spinners providing names of conveyors' deployments has to be refilled by the read related data in the end. The asynchronous task itself initiates a web service invocation by calling the `invokeReadRelatedDataWS()` method of the `WebServiceClient` class. Together with the test venue's ID, a string embodying the name of the target web method ("readConveyorsDeploymentsData") is passed into it. The `invokeReadRelatedDataWS()` method initiates the following execution sequence: firstly, an `org.ksoap2.serialization.SoapObject` request is formed by adding ID of the selected test venue as its property; secondly, the request is added to an `org.ksoap2.serialization.SoapSerializationEnvelope` envelope as an output SOAP object; finally, the target web service is invoked by calling the `call()` method of the `org.ksoap2.transport.HttpTransportSE` class and passing the envelope and the formed string of a SOAP action ("http://pkg/readConveyorsDeploymentsData") that embodies the name of the target web method into it. From this point of the execution flow the server is part of the operation. On the server side the `readConveyorsDeploymentsData()` method of the `CommunicationWithDatabase` class is started to be executed. The method reads those instances of the `ConveyorDeployment` DB entity, which `terrainId` foreign key corresponds to the ID of the selected test venue provided by the requesting client. The read instances of the

[10]See sequence diagram presented in Figure C.2 in appendices.

related conveyors' deployments are returned to the client in the form of the SOAP object. From the obtained SOAP object the `wsInvocationList()` method of the `WebServiceClient` client's class forms a list of conveyors' deployments. The list in the end comes to the `AsyncCallReadConveyorsDeploymentsDataWS` asynchronous task that transfers it to the `MainActivity` by calling its `conveyorsDeploymentsDataHasBeenGot()` method and passing the list of obtained conveyors' deployments and the ID of the spinner that has initiated the reading process into it. If the list of obtained conveyors' deployments is not empty, then their names are presented in the corresponding dropdown list of conveyors' deployments. Otherwise, the dropdown list is filled out by the "No available deployments" text.

4.4.3 Persisting data requests

The persisting data requests (or "Write data to DB" scenario) take place all over the system's execution flows: during the setup of experiments described in detail in Section 4.2.1, during the scan procedure, which logic and execution flows are provided in Sections 4.2.2 and 4.4.4 correspondingly as well as during the fingerprints' acquisition procedure, which logic and execution flows are described in Sections 4.2.3 and 4.4.5 respectively. The setup of experiments implies sending of configured parameters to the server. In the frame of the scan procedure and during the fingerprints' acquisition procedure correspondingly scan results and fingerprints are forwarded to the server.[11]

During forwarding data from a client to the server the "Write data to DB" scenario passes the same steps as the "Read related data from DB" one described at length in Section 4.4.2. First of all, in order to invoke a target web service an instance of the implemented asynchronous task is created and started to be executed. In the following step, the sequence of the web service's invocation is initiated by the asynchronous task. The call of the target web service is performed by a method of the `WebServiceClient` class. As parameters, the formed string of a SOAP action embodying the name of the target web method and the preliminary formed envelope that in this case carries data needed to be written to the DB are passed to the calling method. Finally, the target method of the `CommunicationWithDatabase` class is called on the server side. The method performs necessary operations with the DB: in this scenario it, if necessary, reads parent data that is needful for establishing relationship between the tables, forms related child object and writes it to the DB. As a response, the server forms a string representing data that has been written to the DB. If an error occured while writing, the client that attempted to write the data informs a user about the error by means of a toast. Otherwise, execution flow continues.[12]

[11] Sequence diagram illustrating persisting of data by the example of writing data of a configured nodes' deployment is depicted in Figure C.3 in appendices.

[12] For instance, after successful writing of a nodes' deployment, the client requests the server for the renewed data of nodes' deployments and starts to send data of configured anchors and reference nodes to it. This situation is illustrated in sequence diagram depicted in Figure C.3 in appendices.

4.4.4 Execution flows of the scan procedure and of the subsequent tag's estimated location reading

The scan procedure is performed by the clients of the system. Logic of the scan procedure is described in detail in Section 4.2.2. Below execution flows of the scan procedure and of the subsequent tag's estimated location reading are going to be described.[13]

The core idea of the scan procedure consists in a cyclic execution of the `startScan` and the `stopScan` `java.lang.Runnable`s.[14] The first one is responsible for obtaining of scan results from the available BLE devices (reference nodes and a tag). The final objective of the second one consists in sending of a finally formed tag's scan result to the server.

The `startScan` runnable is sequentially performed specified number of times (namely, 100 times). Thus, accumulation of scan results is organized. After the accumulation, the `stopScan` runnable starts to execute its logic. It filters the accumulated scan results of every detected BLE device (a reference node or a tag) by their RSSIs. Based on the filtered scan results the new tag's scan result is formed and sent to the server in order to be written to the DB.

The procedure of sending of the formed tag's scan result to the server works in accordance with the "Write data to DB" scenario described in Section 4.4.3. After successful writing of the tag's scan result, the grid-scan algorithm is started to be executed. It calculates the tag's estimated location based on the scan results received from the clients of the system. As a result, a `TagEstimatedLocation` object is formed and written to the DB. Implementation logic of the grid-scan algorithm is provided in Section 4.3.2.

The server responses to the client that has initiated the writing. If no error occured while writing to DB, the client continues to operate and starts to read the last tag's estimated location from the server. The reading sequence in general corresponds to the rules of the "Read related data from DB" scenario described in detail in Section 4.4.2. The difference lies in necessity to obtain a single and the last instance of the corresponding DB entity and not a collection of related instances.

If the last tag's estimated location is successfully read, the client shows it graphically to a user. It also checks the `maxNumOfEstimationsReached` flag of the read tag's estimated location. If the flag is set to `true`, the client informs a user about necessity to stop the scan.

[13]Sequence diagram illustrating the scan procedure and the subsequent tag's estimated location reading is presented in Figure C.4 in appendices.

[14]The cyclic execution of the runnables is represented by the outer loop of the sequence diagram depicted in Figure C.4 in appendices. The inner loop represents accumulation of scan results for their further filtering.

4.4.5 Execution flows of the fingerprints' acquisition procedure

The fingerprints' acquisition procedure is performed by the clients of the system. Every client uses fingerprints that it has acquired as additional reference points later on. Theoretical basis of the fingerprints' acquisition procedure is described in detail in Section 3.4. Details of its implementation are provided in Section 4.2.3. Below execution flows of the fingerprints' acquisition procedure are going to be described.[15]

The core idea of the fingerprints' acquisition procedure coincides with the core idea of the scan procedure and consists in a cyclic execution of the `startScan` and the `stopScan` `java.lang.Runnable`s. The first one scans for the particular tag (BLE beacon) that is moving along a powered conveyor (or on a powered line-following robot along its motion path) and obtains a scan result (RSSI, address of the tag and other concomitant data) from it. The second one calculates theoretical coordinates of the tag at the moment of the BLE signal's emission, forms a fingerprint that represents the obtained RSSI that is anchored to the calculated coordinates and sends the fingerprint to the server. Since, implemented filter cannot be applied in the case of fingerprints' acquisition (see basis provided in the frame of the filter's theoretical introduction in Section 3.5), the cyclic execution of the runnables is simplified in this case: the scan results' accumulation loop that is present in logic of the scan procedure (see Figure C.4) is absent here.

When a scan result is obtained, a timestamp is read. The `stopScan` runnable always checks whether the tag is theoretically moving along the conveyor (or the line-following robot's motion path) or the conveyor (the motion path) has been already theoretically passed. The check is performed by comparison of the timestamp with calculated end time of the fingerprints' acquisition procedure. If the timestamp does not exceed the calculated end time, the scan result is considered to be obtained from the tag on its motion path. Then a fingerprint is formed and sent to the server. Moreover, if current time of the client exceeds the calculated end time, the client immediately informs a user about necessity to stop the fingerprints' acquisition procedure by means of a toast.

The sending of the formed fingerprint to the server works in accordance with the "Write data to DB" scenario described in Section 4.4.3. If an error occures while writing to DB, the client signals about the error by means of toast.

[15]Sequence diagram illustrating the fingerprints' acquisition procedure is depicted in Figure C.5 in appendices.

5 System evaluation

The system evaluation aims at determining of localization accuracy of the proposed localization strategies (see their theoretical description provided in Section 3.3). For this purpose several sessions of experiments have been conducted. During the sessions of experiments four from the five proposed localization strategies (namely, the localization strategies 1–4) have been examined.

For each of the strategies a tag has been placed sequentially at several predefined points of the test venue. These are called actual locations of the tag in this thesis. In order to have comparable results, the same set of the tag's actual locations has been considered for testing localization strategies within a particular test venue.

For every actual location of the tag several localization rounds have been performed by the system. As a result a set of estimated locations with calculated localization errors has been formed for each of the examined actual location of the tag. After testing of a localization strategy, mean localization errors have been calculated both for every actual location of the tag and for the test venue, where the localization strategy was tested, as a whole.

The greater number of localization rounds are performed for an actual location of the tag, the more reliable its mean localization error is considered to be and the higher weight it possesses. The weights of the examined actual locations of the tag are taken into consideration, when the mean localization error of the test venue as a whole is calculated. In order to equalize contribution of the mean localization errors of the tag's actual locations to the mean localization error of the test venue as a whole, the principle of equal weights for actual locations of the tag has been introduced (see theory provided in Section 5.2) and implemented (see its application as a part of the grid-scan algorithm described in Section 4.3.2).

The outline of this chapter is as follows: Section 5.1 introduces mathematical basis for the evaluation of the system; Section 5.2 describes the principle of equal weights for the examined actual locations of the tag; Section 5.3 is devoted to description of the conducted experiments and to detailed analysis of their results.

5.1 Levels of evaluation

Two levels of evaluation are proposed for the implemented ILS:

1. Evaluation on the level of actual locations of the tag.
2. Evaluation on the level of localization strategies applied to some particular test venue.

5.1.1 Level of actual locations of the tag

Evaluation on the level of actual locations supposes calculation of mean localization error $\bar{\varepsilon}\,[m]$ as well as of sample standard deviation $s_\varepsilon\,[m]$ of localization errors of every actual location of the tag.

The mean localization error $\bar{\varepsilon}\,[m]$ can be calculated according to Equation (5.1) that represents simple arithmetic average of localization errors calculated by the system while locating the tag placed at the considered actual location:

$$(5.1)\quad \bar{\varepsilon} = \frac{\sum_{i=1}^{n_{est}} \varepsilon_i}{n_{est}},$$

where n_{est} is a number of estimations that have been performed for the considered actual location of the tag and $\varepsilon_i\,[m]$ is an error of the i^{th} estimation.

Error $\varepsilon\,[m]$ of every estimation is composed by its x- and y-components ($\varepsilon_x\,[m]$ and $\varepsilon_y\,[m]$ correspondingly):

$$(5.2)\quad \varepsilon = \sqrt{{\varepsilon_x}^2 + {\varepsilon_y}^2}.$$

Each of the components is calculated as difference of x- or, correspondingly, y-coordinates of the tag's estimated and actual locations.

The mean absolute x-error $\bar{\varepsilon}_{abs_x}\,[m]$ and the mean absolute y-error $\bar{\varepsilon}_{abs_y}\,[m]$ can be calculated according to Equations 5.3 and 5.4 correspondingly:

$$(5.3)\quad \bar{\varepsilon}_{abs_x} = \frac{\sum_{i=1}^{n_{est}} |\varepsilon_{x_i}|}{n_{est}}$$

$$(5.4)\quad \bar{\varepsilon}_{abs_y} = \frac{\sum_{i=1}^{n_{est}} |\varepsilon_{y_i}|}{n_{est}},$$

where n_{est} is a number of estimations that have been performed for the considered actual location of the tag and $\varepsilon_{x_i}\,[m]$ and $\varepsilon_{y_i}\,[m]$ are correspondingly x- and y-components of error $\varepsilon_i\,[m]$ of the i^{th} estimation.

Since it is practically impossible to obtain all the scan results for any possible actual location of the tag, the obtained ones and, hence, localization errors calculated by the system based on them represent data samples for the particular test venue. The sample standard deviation $s_\varepsilon\,[m]$ of localization errors calculated by the system while locating the tag placed at the considered actual location can be calculated according to Equation (5.5) [23]:

$$(5.5)\quad s_\varepsilon = \sqrt{\frac{1}{n_{est}-1}\sum_{i=1}^{n_{est}}(\varepsilon_i - \bar{\varepsilon})^2},$$

where n_{est} is a number of estimations that have been performed for the considered actual location of the tag, $\varepsilon_i\,[m]$ is an error of the i^{th} estimation, $\bar{\varepsilon}\,[m]$ is the mean localization error of the considered actual location of the tag calculated according to Equation (5.1). If distribution of the errors is considered to be approximately normal, then one can say that about 68% of the errors of the particular actual location of the tag fall in the interval of $\bar{\varepsilon} \pm s_\varepsilon\,[m]$.

The calculated mean localization errors as well as sample standard deviations form the data set for higher level of the system's evaluation that is described in Section 5.1.2.

5.1.2 Level of localization strategies

Evaluation on the level of localization strategies represents a more general level of evaluation. This level operates with mean localization errors $\bar{E}\,[m]$ and with sample standard deviations $s_E\,[m]$ of localization errors of the whole test venues. The mean localization error $\bar{E}\,[m]$ can be calculated according to Equation (5.6):

$$(5.6)\quad \bar{E} = \frac{\sum_{i=1}^{N_{act}}(n_{est_i}\bar{\varepsilon}_i)}{\sum_{i=1}^{N_{act}} n_{est_i}},$$

where N_{act} is a number of actual locations of the tag that have been examined within a test venue, $\bar{\varepsilon}_i\,[m]$ is a mean localization error of the i^{th} actual location calculated with the help of Equation (5.1), n_{est_i} is a number of estimations that have been performed for the i^{th} actual location of the tag.

The sample standard deviation $s_E\,[m]$ of localization errors of a test venue as a whole can be calculated according to Equation (5.7), which idea coincides with Equation (5.5) representing the sample standard deviation of localization errors of an actual location of the tag:

$$(5.7)\quad s_E = \sqrt{\frac{1}{N_{est}-1}\sum_{i=1}^{N_{est}}(\varepsilon_i - \bar{E})^2},$$

where $N_{est} = \sum_{i=1}^{N_{act}} n_{est_i}$ is a total number of estimations performed for testing a localization strategy in the particular test venue (N_{act} is a number of actual locations of the tag that have been examined within the test venue, n_{est_i} is a number of estimations performed for the i^{th} actual location of the tag), $\varepsilon_i\,[m]$ is an error of the i^{th} estimation, $\bar{E}\,[m]$ is a mean localization error of the test venue as a whole calculated according to Equation (5.6).

Equation (5.7) can be transformed into Equation (5.8):

$$(5.8)\quad (N_{est} - 1)\, s_E{}^2 = \sum_{i=1}^{N_{act}} \left[\sum_{j=1}^{n_{est_i}} \left(\varepsilon_{ij} - \bar{E}\right)^2 \right],$$

where $\varepsilon_{ij}\,[m]$ is an error of the j^{th} estimation performed for the i^{th} actual location of the tag.

The inner sum of Equation (5.8) can be transformed as follows:

$$(5.9)\quad \begin{aligned} \sum_{j=1}^{n_{est_i}} \left(\varepsilon_{ij} - \bar{E}\right)^2 &= \sum_{j=1}^{n_{est_i}} \left(\left(\varepsilon_{ij} - \bar{\varepsilon}_i\right) + \left(\bar{\varepsilon}_i - \bar{E}\right) \right)^2 \\ &= \sum_{j=1}^{n_{est_i}} \left(\varepsilon_{ij} - \bar{\varepsilon}_i\right)^2 + n_{est_i}\left(\bar{\varepsilon}_i - \bar{E}\right)^2 \\ &= \left(n_{est_i} - 1\right) s_{\varepsilon_i}{}^2 + n_{est_i}\left(\bar{\varepsilon}_i - \bar{E}\right)^2, \end{aligned}$$

where $\bar{\varepsilon_i}\,[m]$ is a mean localization error of the i^{th} actual location of the tag and $s_{\varepsilon_i}\,[m]$ is a sample standard deviation of localization errors calculated for the i^{th} actual location.

Equation (5.8) becomes:

$$(5.10)\quad (N_{est} - 1)\, s_E{}^2 = \sum_{i=1}^{N_{act}} \left[\left(n_{est_i} - 1\right) s_{\varepsilon_i}{}^2 + n_{est_i}\left(\bar{\varepsilon}_i - \bar{E}\right)^2 \right].$$

Then the sample standard deviation $s_E\,[m]$ of a test venue as a whole can be calculated according to the following Equation (5.11):

$$(5.11)\quad s_E = \sqrt{\frac{\sum_{i=1}^{N_{act}} \left(n_{est_i} - 1\right) s_{\varepsilon_i}{}^2 + \sum_{i=1}^{N_{act}} n_{est_i}\left(\bar{\varepsilon}_i - \bar{E}\right)^2}{N_{est} - 1}},$$

where N_{act} is a number of actual locations of the tag that have been examined within the test venue, n_{est_i} is a number of estimations performed for the i^{th} actual location of the tag, $s_{\varepsilon_i}\,[m]$ is a sample standard deviation of localization errors calculated for the i^{th} actual location, $\bar{\varepsilon_i}\,[m]$ is a mean localization error of the i^{th} actual location of the tag, $\bar{E}\,[m]$ is a mean localization error of the test venue as a whole and $N_{est} = \sum_{i=1}^{N_{act}} n_{est_i}$ is a total number of estimations performed for testing a localization strategy in the particular test venue (see Equations 5.1, 5.5 and 5.6).

Sample of localization errors of a test venue as a whole contains subsamples of localization errors of the tag's actual locations that have been examined within the test venue. Equation (5.11) expresses the sample standard deviation of localization errors of a test venue as a whole based on the subsamples' parameters and is used in this thesis for evaluation on the level of localization strategies. If distribution of the errors is considered to be approximately normal, then one can say that about 68% of the errors of the test venue as a whole fall in the interval of $\bar{E} \pm s_E\ [m]$.

5.2 Principle of equal weights for actual locations of the tag

Equation (5.6) utilizes principle of weights for the mean localization errors $\bar{\varepsilon}_i\ [m]$ of the examined actual locations of the tag. The principle of weights is based on the following statement: *the higher number of estimations is performed for an actual location of the tag, the more reliable its mean localization error is considered to be and, as a result, the higher weight it possesses*. In order to disclose the principle of weights underlying calculation of mean localization error $\bar{E}\ [m]$ of a test venue as a whole, Equation (5.6) can be transformed into Equation (5.12):

$$(5.12)\quad \bar{E} = \frac{\sum_{i=1}^{N_{act}} (n_{est_i}\bar{\varepsilon}_i)}{\sum_{i=1}^{N_{act}} n_{est_i}} = \sum_{i=1}^{N_{act}} \left(\frac{n_{est_i}}{\sum_{i=1}^{N_{act}} n_{est_i}} \cdot \bar{\varepsilon}_i \right) = \sum_{i=1}^{N_{act}} (w_i \cdot \bar{\varepsilon}_i)\,,$$

where in addition to the variables exploited in Equation (5.6) a new variable $w_i = \frac{n_{est_i}}{\sum_{i=1}^{N_{act}} n_{est_i}}$ has been introduced. The new variable w_i represents a weight of the mean localization error $\bar{\varepsilon}_i\ [m]$ of the i^{th} actual location of the tag.

In order to balance impacts of the mean localization errors $\bar{\varepsilon}_i\ [m]$ of actual locations of the tag onto the mean localization error $\bar{E}\ [m]$ of the test venue as a whole, weights w_i of the former ones have to be equalized: $w_i = w, i = 1..N_{act}$. This in fact means equalization of numbers of estimations that are performed for every actual location of the tag: $n_{est_i} = n_{est}, i = 1..N_{act}$.

As a result, Equation (5.6) is transformed into the simple arithmetic average that is expressed by Equation (5.13):

$$(5.13)\quad \bar{E} = \frac{\sum_{i=1}^{N_{act}} (n_{est_i}\bar{\varepsilon}_i)}{\sum_{i=1}^{N_{act}} n_{est_i}} = \frac{\sum_{i=1}^{N_{act}} \bar{\varepsilon}_i}{N_{act}}.$$

The principle of equal weights has been introduced after the first session of experiments, which is considered as a pre-test and proof of concept, and is not described and analyzed in detail in this thesis. For the equalization of numbers of estimations that are performed for every actual location of the tag, the number of estimations is verified in the frame of the grid-scan algorithm performed on the server. Logic of the grid-scan algorithm is described in detail in Section 4.3.2.

5.3 Experiments

The purpose of this section is to document and analyze results of the localization experiments that have been conducted with the help of the implemented ILS. At first, methodological sequence of the conducted experiments is presented and test venues are described. Then an overview of the conducted experiments is provided: their configuration parameters, mean localization errors and sample standard deviations of localization errors on both the level of actual locations of the tag and the level of localization strategies and brief conclusions are given. In the end more detailed analysis of the obtained results is performed.

5.3.1 Methodology

Several sessions of localization experiments have been conducted. The methodology reflects the way how the experiments were approached and the system was evolved: the sequence of the experiments' sessions, test venues, where they have been conducted, localization strategies that have been tested in the each experiments' session, observations that have been done during the experiments and the extensions to the system introduced after every experiments' session.

The methodology is summarized in Table 5.1. Briefly, five session of experiments have been conducted, for which two test venues (namely, meeting room and classroom that is situated in the clean room building) have been used (see description of the test venues provided in Section 5.3.2). Starting from the second session of experiments, six anchors and six reference nodes were exploited. During the sessions of experiments four from the proposed five localization strategies (namely, the localization strategies 1–4) have been tested (see explanation of the strategies provided in Section 3.3). The main extensions to the system introduced during the sessions of experiments include: setup of a dedicated local Wi-Fi network, implementation of the principle of equal weights for the examined actual locations of the tag (see its description provided in Section 5.2), unification of the tag's and the reference nodes' hardware and implementation of an RSSI filter.

Table 5.1: Methodological sequence of the conducted experiments

Session of experiments	Test venue	Number of anchors/ reference nodes	Tested localization strategies	Observations	Extensions to the system introduced after the session
Session 1	meeting room	3/3	1 and 3	1. Unstable behavior of the used shared Wi-Fi network; 2. Unfixed number of estimations performed for every actual location of the tag leading to improper estimations	1. Dedicated local Wi-Fi network; 2. Number of estimations that is fixed and equal for all of the examined actual locations of the tag
Session 2	meeting room	6/6	1 and 3	1. Different hardware of the tag (TI CC2541 SensorTag) and the reference nodes (RedBear Beacon B1); 2. High variance of RSSIs leading to the high variance of accuracy	Reference nodes: replacement of the RedBear Beacons B1 by the TI CC2541 SensorTags
Session 3	classroom in the clean room building	6/6	1 and 3	High variance of RSSIs leading to the high variance of accuracy	Filtering the obtained RSSIs
Session 4	classroom in the clean room building	6/6	1 and 3	Lower variance of accuracy	—
Session 5	classroom in the clean room building	6/6	2 and 4	—	—

5.3.2 Test venues

At different stages the implemented ILS has been tested at the following two venues:

1. meeting room
2. classroom in the clean room building

Parameters of the test venues are provided in Table 5.2.

Name	Width [m]	Length [m]
meeting room	4.75	6.15
classroom in the clean room building	8.55	10.45

Table 5.2: Parameters of the test venues

The *meeting room* is a rectangular room with a chamfered corner. Windows are mounted along the short longitudinal wall. During the experiments desks that are being in the room have been placed in the middle of the room in a shape of a rectangle.

The *classroom in the clean room building* is a rectangular room with windows mounted along one of the longitudinal walls. During the experiments some of the tables that are being in the room have been placed in the middle of the room in a U-shape.

Photos of the test venues are presented in Figure 5.1.

(a) meeting room

(b) classroom in the clean room building

Figure 5.1: Photos of the test venues

5.3.3 Session 1

The first session of experiments has been conducted in the meeting room with tree anchors and three reference nodes. In this session localization strategies 1 and 3 have been tested. The first session of experiments is considered as a proof of concept, therefore its localization results are not listed and analyzed in this thesis. However, while conducting the session the following was observed:

1. The shared Wi-Fi network used for the client-server communication showed unstable behavior. As a result, the clients of the system periodically lost connection to the server and did not provide scan results for their further processing.
2. The number of estimations performed for every actual location of the tag was not fixed and equal for all of the examined actual locations. This led to unequal contribution of mean localization errors of the actual locations of the tag to the mean localization error of the test venue as a whole and, as a result, to improper estimations.

The first problem has been solved by setup of a dedicated local Wi-Fi network that could be deployed at any test venue. In order to overcome the second problem, the principle of equal weights for actual locations of the tag has been introduced (see Section 5.2) and implemented (see its application as a part of the grid-scan algorithm described in Section 4.3.2).

5.3.4 Session 2

The second session of experiments was a repetition of the first one while including the improvements introduced after the first session. As it was expected, the dedicated local Wi-Fi network that has been deployed after the first session of experiments in exchange for the unstable shared Wi-Fi network has shown stable connectivity. After the first session of experiments a counter of estimations performed for every actual location of the tag has been introduced. The counter initiated stop of the server's localization procedure whenever the fixed maximal number of estimations of every examined actual location of the tag has been reached. Thus, all of the examined tag's actual locations had the same number of estimations and, therefore, possessed equal weights, when calculating the mean localization error of the test venue as a whole (see theory provided in Section 5.2). Toasts that popped up on the screens of Android clients indicated about necessity to stop the client's scan procedure.

For execution of the grid-scan algorithm, which core idea is provided in Section 3.2.3 and which implementation details are described in Section 4.3.2, the area of the test venue (of the meeting room) has been broken up into the cells: 15 cells along width and 20 cells long length. As a result, the size of every cell has come to 0.32 m (width) and 0.31 m (length).

This time six anchors and six reference nodes have been used. With such a set of anchors and reference nodes localization strategies 1 and 3 have been tested.

The tag has been represented by the TI CC2541 SensorTag, the reference nodes have been represented by the RedBear Beacons B1. For both types of the BLE beacons the output power was equally set to 0 dBm.

5.3.4.1 Test of localization strategy 1

The localization strategy 1 considers the case, when anchors and reference nodes are pairwise spatially merged with each other. Coordinates of the anchors and reference nodes that took part in the test of the localization strategy 1 are given in Table 5.3. z-coordinates (the altitudes of the deployed anchors and reference nodes) are not considered in the localization procedure. However, in order to document the experiment setup thoroughly the z-coordinates are presented as well.

x [m]	y [m]	z [m]
0,20	0,20	0,92
4,55	0,75	0,73
4,55	2,75	0,73
4,55	5,95	0,00
2,40	5,95	0,45
0,20	3,72	0,92

Table 5.3: Session 2, test of localization strategy 1: coordinates of anchors and reference nodes

In this test of localization strategy 1 nine actual locations of the tag have been examined. The tag has been sequentially placed at different points on the desks that had been pushed together in the middle of the room (see Figure 5.1a). The nine examined actual locations of the tag formed a grid have allowed to cover the middle of the room by the localization tests. For every examined actual location of the tag 50 estimations have been performed.

GUI of the test's configuration is depicted in Figure 5.2. The GUI provides the functionality to select a preliminary created experiment (includes parameters of the test venue, its granulation onto the cells, data of the anchors' and reference nodes' deployment), set up an actual location of the tag, start/stop the scan procedure and represents the complete configuration and (later on) the progress of the localization procedure.[1]

[1]Localization results of this test and of the following test of localization strategy 3 are provided in Table D.1 in appendices.

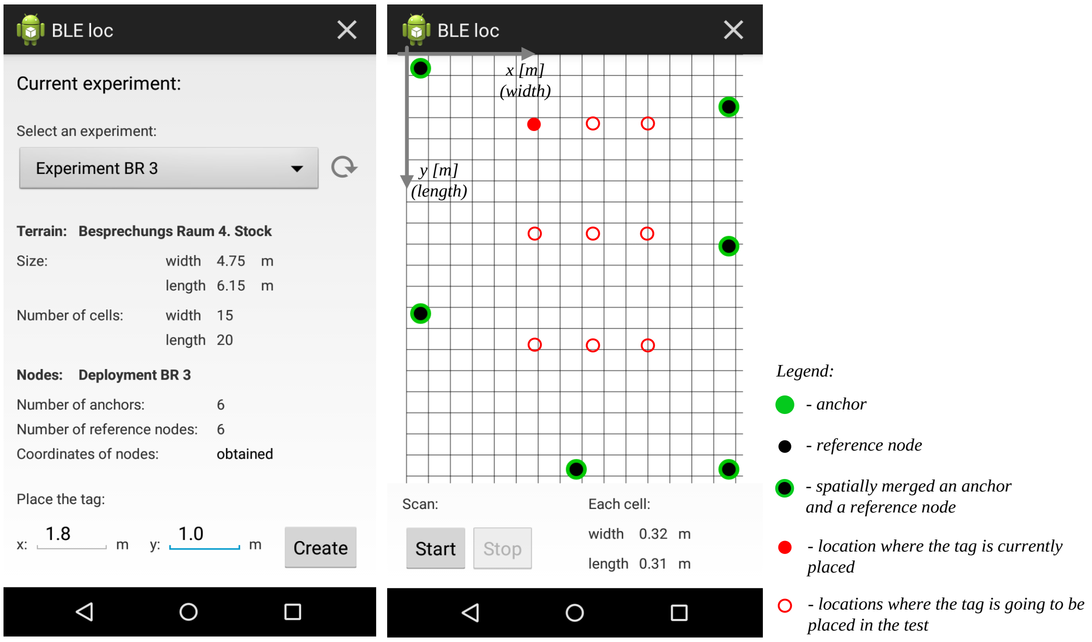


Figure 5.2: Session 2, test of localization strategy 1: GUI of the test's configuration

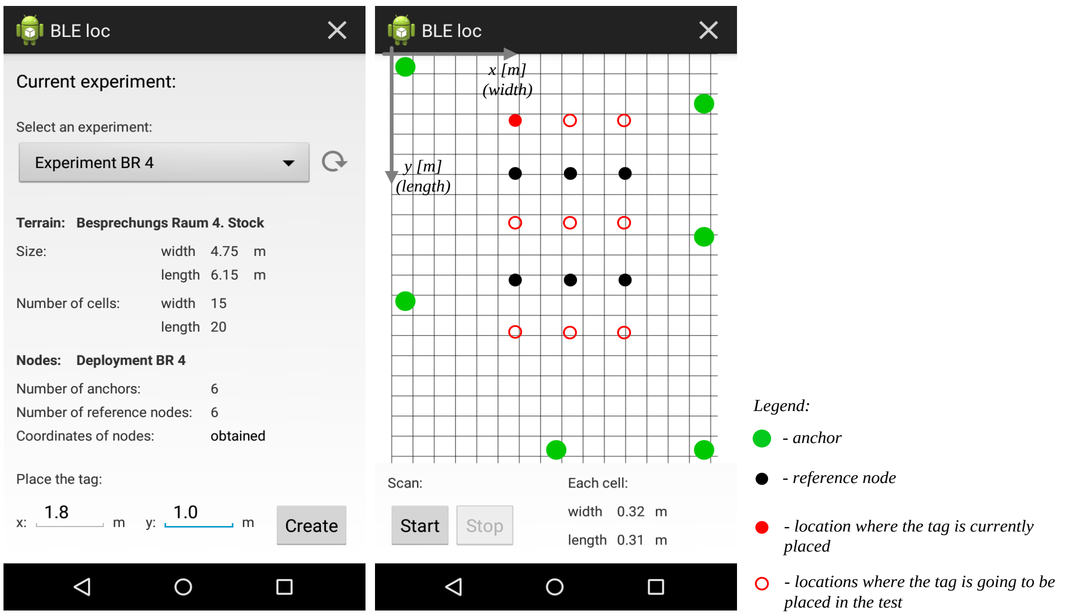


Figure 5.3: Session 2, test of localization strategy 3: GUI of the test's configuration

5.3.4.2 Test of localization strategy 3

The localization strategy 3 considers a spatial distribution of anchors and reference nodes. Therefore, deployment of anchors and reference nodes has been organized as follows: the anchors have stayed where they were during the test of the localization strategy 1, the reference nodes have been placed on the desks in the middle of the room. Coordinates of anchors and reference nodes are provided in Table 5.4.

	x [m]	y [m]	z [m]		x [m]	y [m]	z [m]
Anchors	0,20	0,20	0,92	**Reference nodes**	1,80	1,80	0,73
	4,55	0,75	0,73		2,60	1,80	0,73
	4,55	2,75	0,73		3,40	1,80	0,73
	4,55	5,95	0,00		1,80	3,40	0,73
	2,40	5,95	0,45		2,60	3,40	0,73
	0,20	3,72	0,92		3,40	3,40	0,73

Table 5.4: Session 2, test of localization strategy 3: coordinates of anchors and reference nodes

In order to perform a proper comparison of the localization strategies tested within the same particular test venue, the same actual locations of the tag have always been examined in the particular test venue. In this test of localization strategy 3 the same nine actual locations of the tag (as in the foregoing test of the localization strategy 1) have been examined. For every actual location of the tag 50 estimations have been performed.

Configuration of the test is depicted in Figure 5.3.

5.3.4.3 Observations

The localization strategy 3 has demonstrated better accuracy and lower deviation of localization errors from their mean than the localization strategy 1 (namely, the strategies have resulted in the mean localization errors of 1.43 m and 1.57 m correspondingly and in the sample standard deviations of localization errors of 0.66 m and 0.96 m respectively).[2] However, the results show noticeable fluctuations of estimations within the test venue. The fluctuations arising from the high variety of the obtained RSSIs influence localization accuracy of the system. According to the results, maximal localization error of the tested localization strategy 1 appeared during one of the estimations has come to 5.32 m. The localization strategy 3 has shown the maximal localization error of 4.03 m.

[2]See localization results provided in Table D.1 in appendices.

5.3.5 Session 3

The third session of experiments has been performed in the classroom that is situated in the clean room building. The classroom is three times bigger than the meeting room where the previous sessions of experiments have been conducted (see size parameters of the test venues in Table 5.2). The main interest of this session of experiments consisted in testing the localization strategies within a bigger test venue.

In the frame of the previous two sessions of experiments, the tag and the reference nodes have been represented by different hardware (by the TI CC2541 SensorTag and by the RedBear Beacons B1 respectively). Both types of beacons have possessed the same output power of 0 dBm. However, different types of hardware may have different antenna characteristics. Therefore, for the clarity of experiment the RedBear Beacons B1 have been replaced by the TI CC2541 SensorTags.

The area of the test venue has been broken up into 27 cells along width and 34 cells along length. Thus the same degree of granulation as it has been set up earlier for the second session of experiments has been achieved: size of every cell has come to 0.32 m (width) and 0.31 m (length).

The third session has been conducted with six anchors and six reference nodes. With this set of anchors and reference nodes localization strategies 1 and 3 have been tested.

5.3.5.1 Test of localization strategy 1

Coordinates of anchors and reference nodes participated in this test are given in Table 5.5.

x [m]	y [m]	z [m]
0,60	1,70	0,73
8,55	1,10	0,82
8,55	5,05	0,82
8,55	9,40	0,82
0,20	10,25	0,77
0,20	5,95	0,63

Table 5.5: Sessions 3 and 4, test of localization strategy 1: coordinates of anchors and reference nodes

Nine actual locations of the tag have been examined during the test. The tag has been sequentially placed at different points on the desks forming a U-shape in the middle of the room (see Figure 5.1b) and on the floor inside the U-shape. The nine examined actual locations

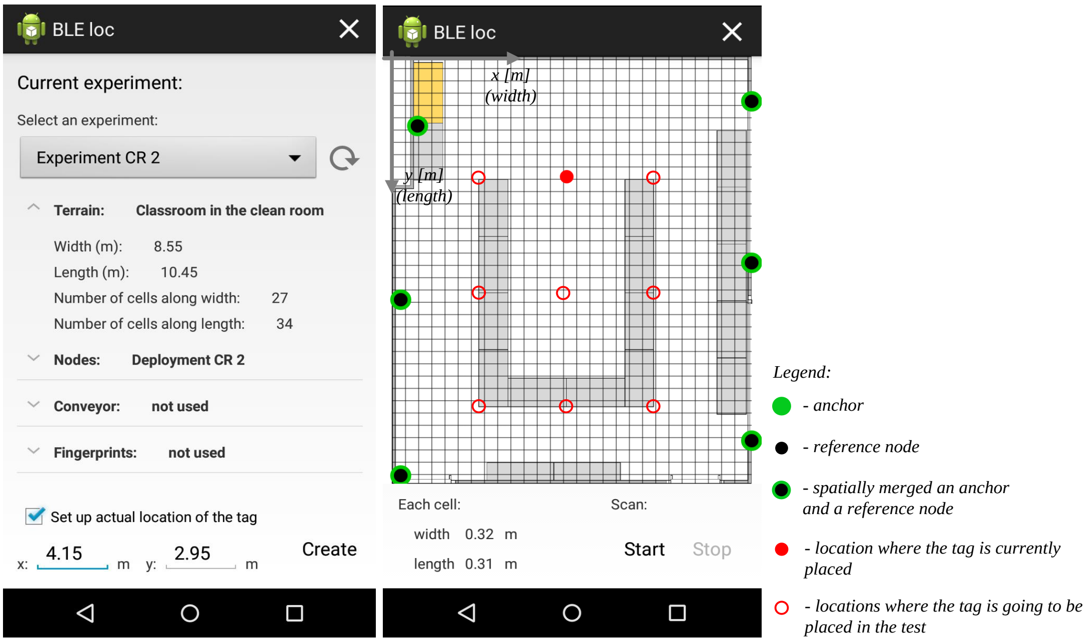


Figure 5.4: Sessions 3 and 4, test of localization strategy 1: GUI of the test's configuration

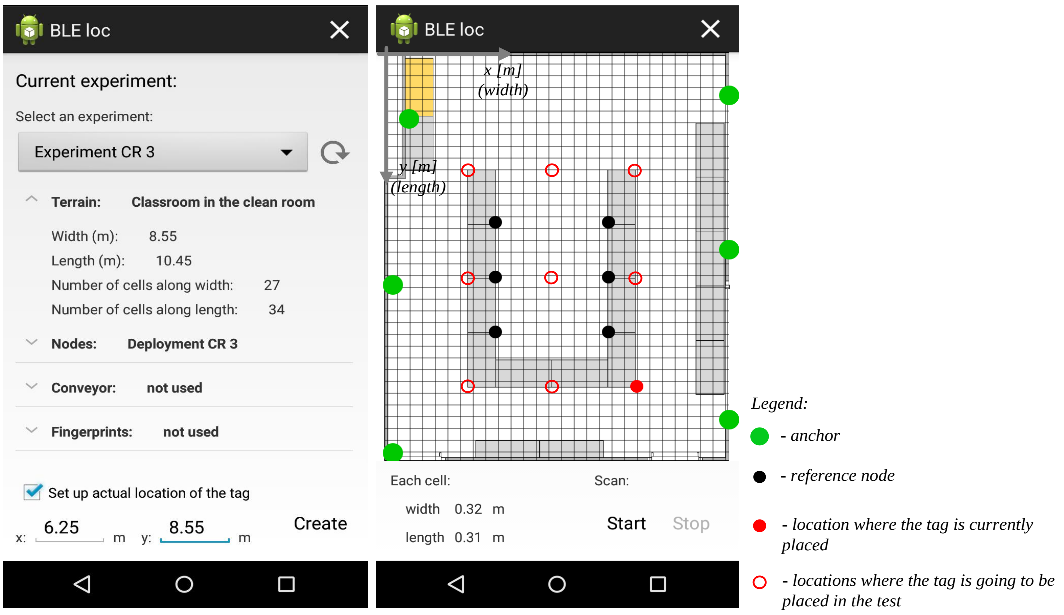


Figure 5.5: Sessions 3 and 4, test of localization strategy 3: GUI of the test's configuration

of the tag formed a grid have allowed to cover the middle of the room by the localization tests. Seven of the actual locations of the tag have been estimated 100 times each. The rest two of them have been estimated 90 times each.

Configuration of the test is depicted in Figure 5.4.[3]

5.3.5.2 Test of localization strategy 3

Coordinates of anchors and reference nodes participated in this test are provided in Table 5.6. After the test of the localization strategy 1 the anchors have stayed where they were. The reference nodes have been placed on the desks forming a U-shape in the middle of the room.

	x [m]	y [m]	z [m]		x [m]	y [m]	z [m]
Anchors	0,60	1,70	0,73	**Reference nodes**	2,75	4,35	0,73
	8,55	1,10	0,82		2,75	5,75	0,73
	8,55	5,05	0,82		2,75	7,15	0,73
	8,55	9,40	0,82		5,55	4,35	0,73
	0,20	10,25	0,77		5,55	5,75	0,73
	0,20	5,95	0,63		5,55	7,15	0,73

Table 5.6: Sessions 3 and 4, test of localization strategy 3: coordinates of anchors and reference nodes

In the test of localization strategy 3 the same nine actual locations of the tag (as in the foregoing test of the localization strategy 1) have been examined. For every actual location of the tag 50 estimations have been performed.

Configuration of the test is depicted in Figure 5.5.

5.3.5.3 Observations

The localization strategy 3 has again demonstrated better accuracy and lower deviation of localization errors from their mean than the localization strategy 1 (namely, the strategies have resulted in the mean localization errors of 2.96 m and 4.05 m correspondingly and in the sample standard deviations of localization errors of 1.38 m and 2.05 m respectively).[4]

[3] Localization results of this test and of the following test of localization strategy 3 are provided in Table D.2 in appendices.

[4] See localization results provided in Table D.2 in appendices.

However, the localization accuracy has decreased in comparison with the previous session of experiments described in Section 5.3.4. The reason lies in the larger size of the test venue while the same number of anchors and reference nodes and similar deployment approach have been used. The change of the test venue has led to longer distances between the anchors and the reference nodes. This, in turn, has influenced the variety of obtained RSSIs and, as a result, the variety of estimations, since the BLE signals have become more affected by the noise on the longer way (see Section 3.5). According to the results, maximal localization error of the tested localization strategy 1 appeared during one of the estimations was 9.19 m. The localization strategy 3 has shown a maximal localization error of 6.15 m. The localization accuracy has been also affected by enlargement of the rings' and circles' intersections, which centers of gravity are considered as estimated locations of the tag (see Section 3.2).

5.3.6 Session 4

Before the forth session of experiments an RSSI filter has been implemented. The filter is intended to eliminate fluctuations in obtained RSSIs and thus increase the localization accuracy of the system. The core idea of the filter is provided in Section 3.5. Its implementation details are described in Section 4.2.2.

In order to evaluate the performance of the implemented filter, the forth session of experiments has been repeated in the classroom that is situated in the clean room building (see Figure 5.1b). In this session of experiments localization strategies 1 and 3 have been tested once again. Deployment of anchors and reference nodes, placing of actual locations of the tag as well as the degree of granulation of the test venue have been replicated from the previous session of experiments.

5.3.6.1 Test of localization strategy 1

For this test anchors and reference nodes have been placed at the same points as for the test of the localization strategy 1 during the previous session of experiments (see Table 5.5). The same nine actual locations of the tag have been examined. At this time all of them have been estimated 100 times each.

Configuration of this test corresponds to the configuration that is depicted in Figure 5.4.[5]

[5] Localization results of this test and of the following test of localization strategy 3 are provided in Table D.3 in appendices.

5.3.6.2 Test of localization strategy 3

For this test anchors and reference nodes have been deployed at the same points as for the test of localization strategy 3 in the previous session of experiments (see Table 5.6). The same nine actual locations of the tag have been examined. For each of them 100 estimations have been performed.

Configuration of the test corresponds to the configuration that is depicted in Figure 5.5.

5.3.6.3 Observations

The implemented filter has yielded good results.[6] The localization accuracy of the system has improved: the localization strategy 1 has resulted in the mean localization error of 3.61 m (compared to its result of 4.05 m in the previous session of experiments), the localization strategy 3 has resulted in the mean localization error of 2.34 m (against its result of 2.96 m in the previous session of experiments). Due to the filtering, the maximal localization errors of the localization strategies 1 and 3 have diminished to 7.44 m and 5.66 m correspondingly. This session of experiments has also shown a decrease in deviation of localization errors from their mean for the localization strategy 1: it has resulted in a sample standard deviation of localization errors of 1.46 m (against 2.05 m in the previous session of experiments). In general, one can say that localization strategy 3 has again shown better performance than localization strategy 1, except for its sample standard deviation of localization errors that has unexpectedly increased to 1.51 m and exceeded the same parameter of the localization strategy 1 by 5 cm.

5.3.7 Session 5

The fifth session of experiments was intended for evaluating the influence of preliminary calibrating the system with the help of an automatic movement (in this particular case, with the help of a line-following robot) on the localization accuracy. Thus, in the fifth session of experiments localization strategies 2 and 4 have been tested (description of the strategies is provided in Sections 3.3.2 and 3.3.4 correspondingly). In order to have an opportunity to evaluate performance of the foregoing localization strategies 2 and 4 in comparison with performance of localization strategies 1 and 3, the fifth session of experiments has been conducted in the classroom that is situated in the clean room building (see Figure 5.1b), where the latter couple of strategies has been already tested during the previous sessions of experiments. Deployment of anchors and reference nodes, placing of actual locations of the tag as well as degree of granulation of the test venue have been replicated from the third

[6] See localization results provided in Table D.3 in comparison with localization results provided in Table D.2 in appendices.

and from the forth sessions of experiments (see their descriptions in Sections 5.3.5 and 5.3.6 correspondingly). The motion path of the line-following robot has been provided along the U-shape that is formed by desks in the middle of the room. Before preliminary calibrating the system (namely, before the fingerprints' acquisition procedure) with the help of the robot, its speed has been measured. It was 0.21 m/s. The robot's motion path consisted of three path sections. Their coordinates are provided in Table 5.7. Altitudes of the path sections (their z-coordinates) are not taken into account in the further localization procedure and are provided in order to document the experiment setup thoroughly. The line-following robot on its motion path with a tag placed onto it is depicted in Figure 5.6. During the fingerprints' acquisition procedure every anchor has acquired 4–5 fingerprints.

	x [m]	**y [m]**		**x [m]**	**y [m]**		**z [m]**
Beginning	5,90	2,95	**End**	5,90	8,20	**Altitude**	0,73
	5,90	8,20		2,40	8,20		0,73
	2,40	8,20		2,40	2,95		0,73

Table 5.7: Session 5: coordinates of the robot's path sections

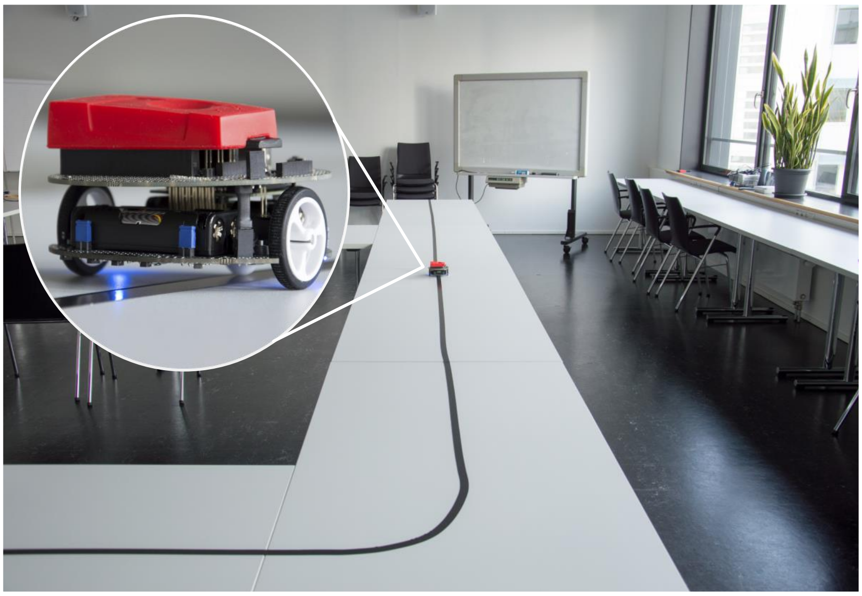


Figure 5.6: Session 5: line-following robot on its motion path with a tag placed onto it

5.3.7.1 Test of localization strategy 2

For this test anchors and reference nodes have been placed at the same points as for the tests of localization strategy 1 during the third and the forth sessions of experiments (see Table 5.5). During the test the tag has been sequentially placed at the same nine actual locations as during the third and the forth sessions of experiments conducted in the classroom. Every actual location of the tag has been estimated 100 times.

Configuration of the test is depicted in Figure 5.7.[7]

5.3.7.2 Test of localization strategy 4

For this test anchors and reference nodes have been placed at the same points as for the tests of localization strategy 3 during the third and the forth sessions of experiments (see Table 5.6). The same nine actual locations of the tag have been chosen for their estimation. Every actual location of the tag has been estimated 100 times.

Configuration of the test is depicted in Figure 5.8.

5.3.7.3 Observations

The tests of the localization strategies 2 and 4 have shown ambiguous results.[8] On the one hand, the test of the localization strategy 2 has confirmed expectations from usage of an automatic movement (in this case, of the line-following robot) for the purpose of preliminary calibration of the system. Localization strategy 2 has resulted in the mean localization error of 2.67 m. For comparison, localization strategy 1 that operated with the same deployment of nodes, but did not leverage an automatic movement, has resulted in the mean localization error of 3.61 m during the previous session of experiments. On the other hand, the test of the localization strategy 4 has not shown any accuracy improvements in comparison with the localization strategy 3 that operated with the same deployment of nodes, but did not use an automatic movement for the preliminary calibration of the system. The localization strategy 4 has resulted in the mean localization error of 2.36 m, whereas the localization strategy 3 tested in the frame of the previous session of experiments has resulted in 2.34 m. However, the localization strategy 4 has shown lower deviation of localization errors from their mean than the localization strategy 3: their sample standard deviations of localization errors have come to 1.07 m and 1.51 m correspondingly. The localization strategy 2, on the contrary, has not resulted in a lower deviation of localization errors from their mean in comparison with

[7] Localization results of this test and of the following test of localization strategy 4 are provided in Table D.4 in appendices.

[8] See localization results provided in Table D.4 in comparison with localization results provided in Table D.3 in appendices.

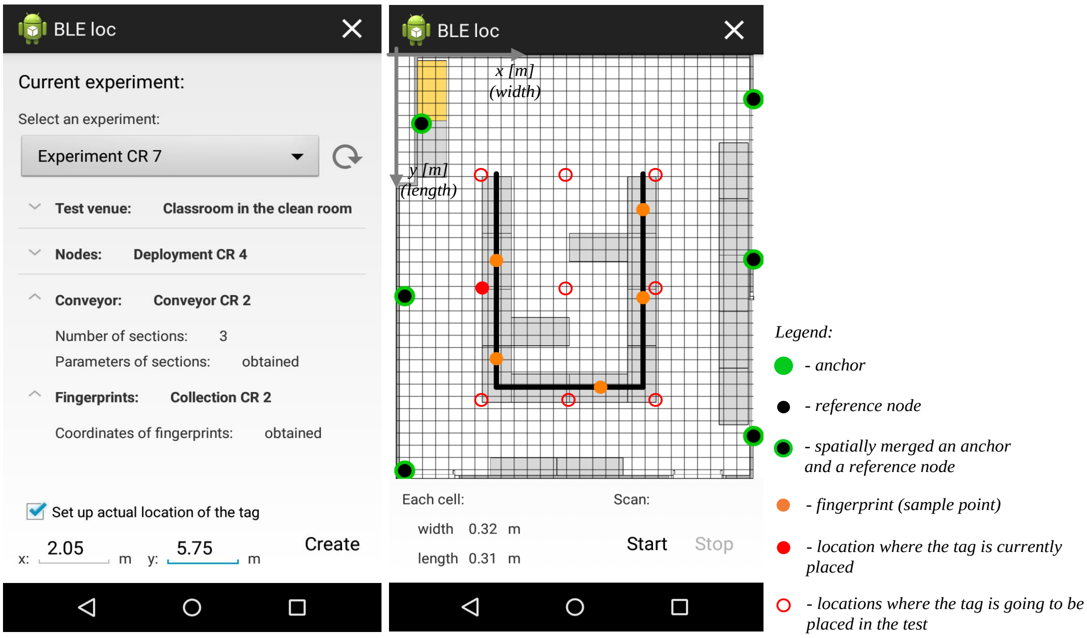


Figure 5.7: Session 5, test of localization strategy 2: GUI of the test's configuration

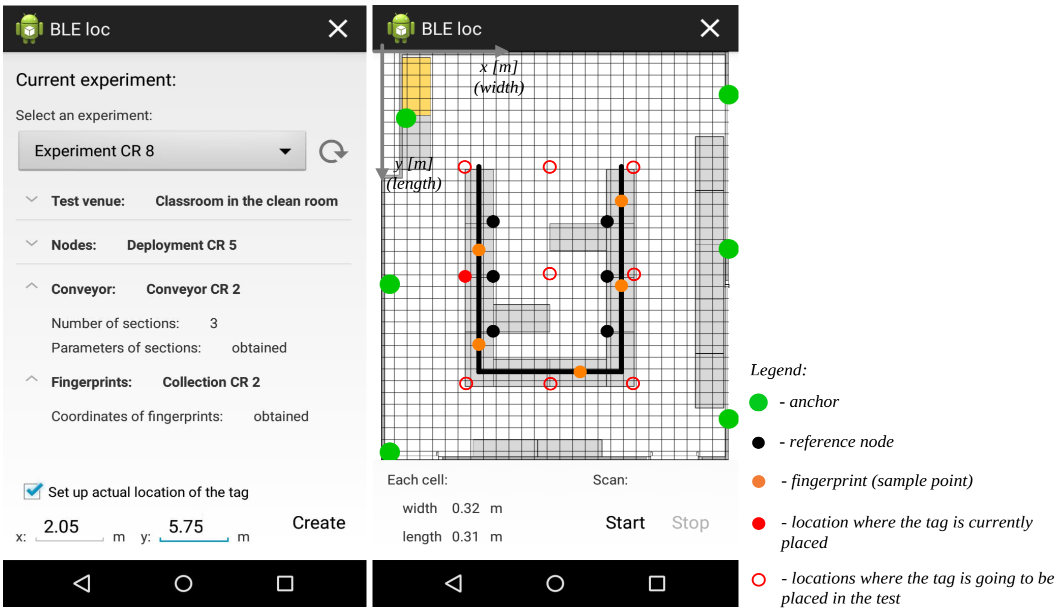


Figure 5.8: Session 5, test of localization strategy 4: GUI of the test's configuration

the localization strategy 1. In order to confirm positive effect of the preliminary calibration of the system with the help of an automatic movement, repeated sessions of experiments are needed. However, the fifth session of experiments has confirmed the following tendency once again: the considered deployments of anchors and reference nodes, where they are spatially distributed with respect to each other, show better localization performance than the considered deployments, when every anchor is spatially merged with a reference node.

5.3.8 Evaluation of results

This section provides detailed analysis of the localization results obtained during the sessions of experiments described in Sections 5.3.4, 5.3.5, 5.3.6 and 5.3.7.[9] The following approaches are proposed for the analysis of the localization results:

1. *Comparison of localization strategies tested in every particular session of experiments:* Since every particular session of experiments considered a test of localization strategy working with spatially merged anchors and reference nodes and a test of localization strategy considering their spatially distributed deployment, this approach to evaluation of results aims at comparison of the implemented ILS's performance for the cases, when anchors and reference nodes are spatially merged and when they are spatially distributed.
2. *Comparison of localization strategies of the same name tested in the third and in the forth sessions of experiments:* Since the forth session of experiments differs from the third one in filtering of obtained RSSIs, this approach is intended for assessment of the filter's performance.
3. *Comparison of the localization strategies 1 and 2, and the localization strategies 3 and 4 tested in the forth and in the fifth sessions of experiments correspondingly:* Since the localization strategies 1 and 2 (as well as the localization strategies 3 and 4) differ from each other only in absence or presence of a preliminary calibration of the system with the help of an automatic movement, this approach to evaluation of results is intended for assessment of the implemented ILS's performance without and with usage of an automatic movement (in this particular case, of a line-following robot) for the system's preliminary calibration.

5.3.8.1 Comparison of localization strategies tested in every particular session of experiments

During every particular session of experiments tests of two different localization strategies have been conducted. The first of the two tested localization strategies has always considered spatially merged anchors and reference nodes (localization strategies 1 and 2). The second

[9]The obtained localization results that are going to be graphically presented and evaluated in this section are provided in Tables D.1, D.2, D.3 and D.4. Summary of the results is provided in Table D.5 in appendices.

one has always worked with their spatially distributed deployment (localization strategies 3 and 4).

Localization results of tests conducted in two different test venues (namely, in the meeting room and in the classroom that is situated in the clean room building) are depicted in Figures 5.9 and 5.10. The figures represent evaluation of performance of the implemented ILS on the level of localization strategies (mathematical basis of this evaluation level is provided in Section 5.1.2). The figures show mean localization errors $\bar{E}\,[m]$ and sample standard deviations $s_E\,[m]$ of localization errors of the tested localization strategies for every test venue as a whole.

Figures 5.9 and 5.10 clearly show that each of the tested localization strategies working with spatially distributed deployment of anchors and reference nodes (namely, the localization strategy 3 or 4) excels the one that has been tested in the same session of experiments and that works with spatially merged deployment of anchors and reference nodes (namely, the localization strategy 1 or 2) in localization accuracy and in deviation of localization errors from their mean. The only one session of experiments, when the spatially distributed deployment of anchors and reference nodes has resulted in a slightly higher deviation of localization errors from their mean (but still in a better localization accuracy) than the spatially merged one, was the forth session of experiments.[10]

More detailed comparison of localization strategies tested in the frame of every particular session of experiments can be performed on the level of actual locations of the tag (mathematical basis of this evaluation level is provided in Section 5.1.1). Graphs that provide such kind of comparison are depicted in Figures 5.11, 5.12, 5.13 and 5.14. Each of the graphs shows nine actual locations of the tag examined in the frame of a session of experiments with the help of two localization strategies. For each of the nine actual locations of the tag mean localization errors $\bar{\varepsilon}\,[m]$ and sample standard deviations $s_\varepsilon\,[m]$ of localization errors of the tested localization strategies are presented. The graphs show that the localization strategy 3 working with a spatially distributed deployment of anchors and reference nodes has resulted in better localization accuracy for most of the examined actual locations of the tag and in evidently lower deviation of localization errors from their means than the localization strategy 1 considering spatially merged anchors and reference nodes (see Figures 5.11, 5.12 and 5.13). Despite the fact that for a test venue as whole the localization strategy 4 (spatially distributed anchors and reference nodes) has resulted in a higher localization accuracy and in a lower deviation of localization errors from their mean than the localization strategy 2 (spatially merged anchors and reference nodes)[11], their comparison on the level of actual locations of the tag (see Figure 5.14) does not reflect this fact evidently. The localization strategies 2 and 4 that leverage an automatic movement for preliminary calibration of the system should be tested additionally.

[10]See localization results provided in Table D.3 in appendices.

[11]See Table D.4 provided in appendices as well as Figure 5.10.

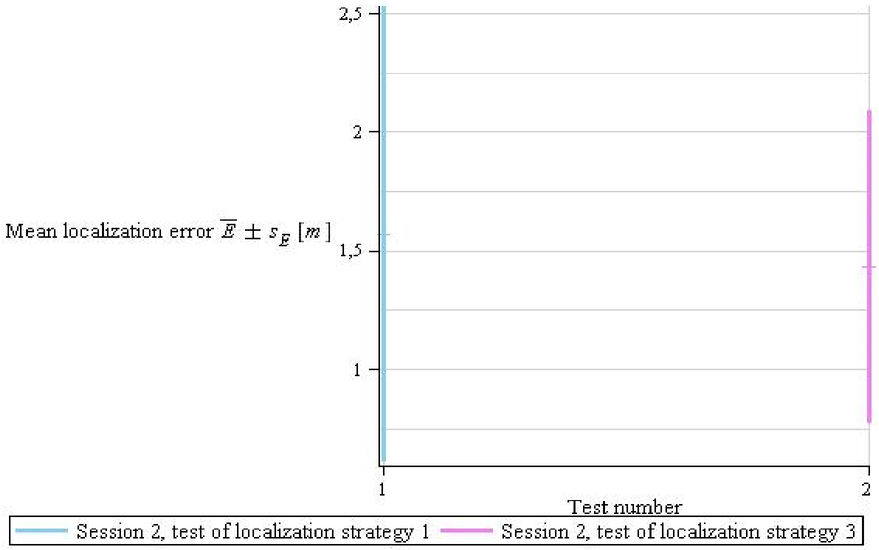


Figure 5.9: Localization results of tests conducted in the meeting room

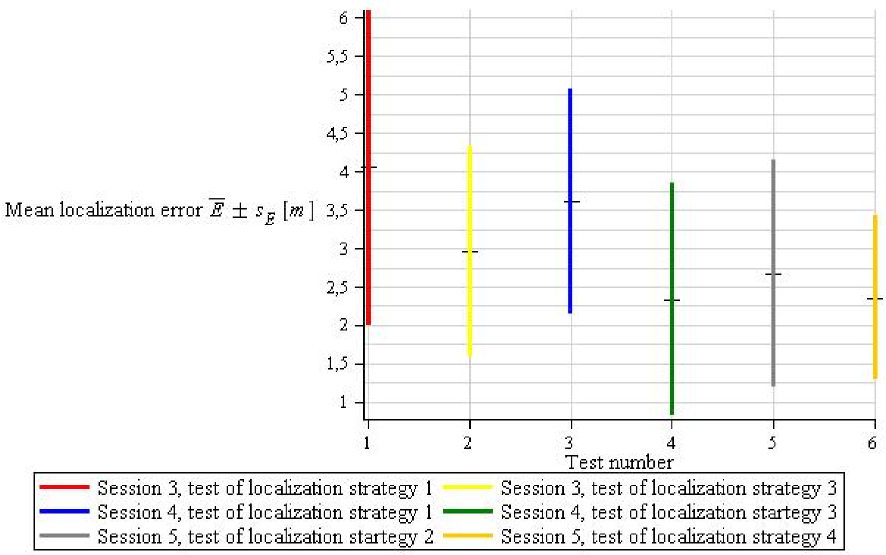


Figure 5.10: Localization results of tests conducted in the classroom that is situated in the clean room building

Nevertheless, in general, the comparison of localization strategies tested in every particular session of experiments (in fact, the comparison of implemented ILS's performance for the cases, when anchors and reference nodes are spatially merged and when they are spatially distributed) has shown advantages of the latter approach.

5.3.8.2 Comparison of localization strategies of the same name tested in the *third* and in the *forth* sessions of experiments

After the third session of experiments that was conducted in the classroom that is situated in the clean room building and that tested the localization strategies 1 and 3, an RSSI filter has been implemented. In order to assess its performance, the same two localization strategies have been tested once again in the forth session of experiments.

Localization results of the tested strategies reflecting their performance in the test venue as a whole are presented in Figure 5.10. The results show that localization accuracy of the strategies tested in the forth session of experiments is higher than localization accuracy of the strategies having the same name and tested in the third session of experiments.[12] More detailed comparison of localization strategies of the same name tested in the third and in the forth sessions of experiments is provided in Figures 5.15 and 5.16. The comparison is performed on the level of actual locations of the tag. Even though Figure 5.15 does not provide an obvious picture of improvement in localization accuracy for the localization strategy 1 with introduction of the RSSI filter, Figure 5.16 evidently confirms such kind of improvement for the localization strategy 3.

Figures 5.15 and 5.16 also show a reduction in deviation of localization errors from their means with the filter's introduction. Although the test of the localization strategy 3 in the fourth session of experiments has not shown reduction in this parameter for the test venue as a whole in comparison with the test of the same strategy in the third session of experiments[13], the former one has shown an obvious reduction in deviation of localization errors from their means in comparison with the latter one on the level of actual locations of the tag (see Figure 5.16). The slightly higher deviation of this parameter that the former one has shown for the test venue as a whole in comparison with the latter one can be explained by strong deteriorations in its localization accuracy for the first and for the ninth actual locations of the tag. Such kind of fluctuations have influenced mean localization error of the test venue as a whole, which itself has made an impact on sample standard deviation of localization errors of the test venue as a whole (see Equations 5.6 and 5.11).

In general, one can confidently state that the implemented RSSI filter has improved localization accuracy of the system and decreased deviation of localization errors from their mean.

[12] See also localization results provided in Tables D.2 and D.3 in appendices.
[13] See Tables D.2 and D.3 provided in appendices as well as Figure 5.10.

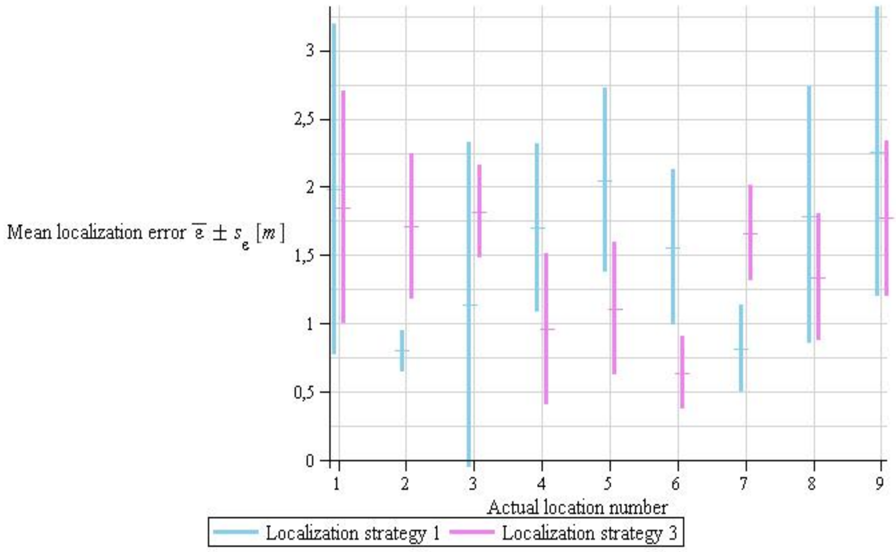


Figure 5.11: Performance of localization strategies 1 and 3 tested in the *second* session of experiments

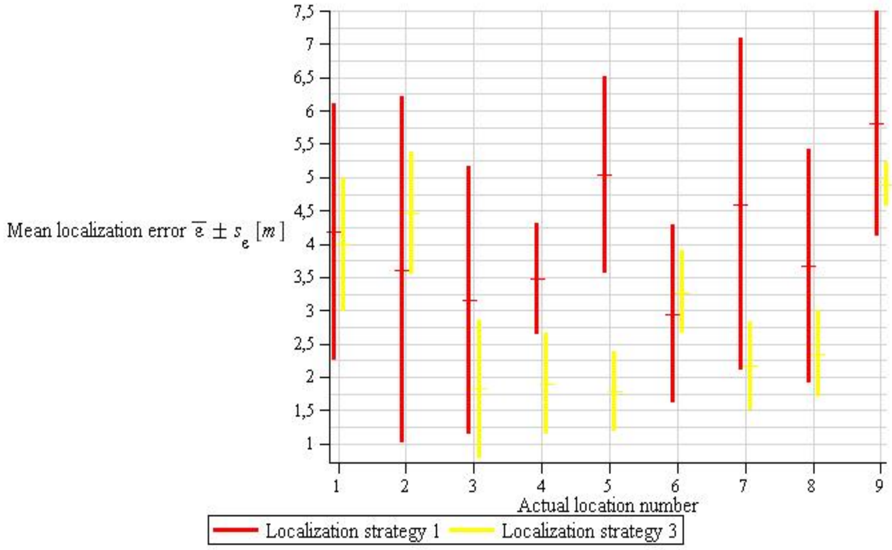


Figure 5.12: Performance of localization strategies 1 and 3 tested in the *third* session of experiments

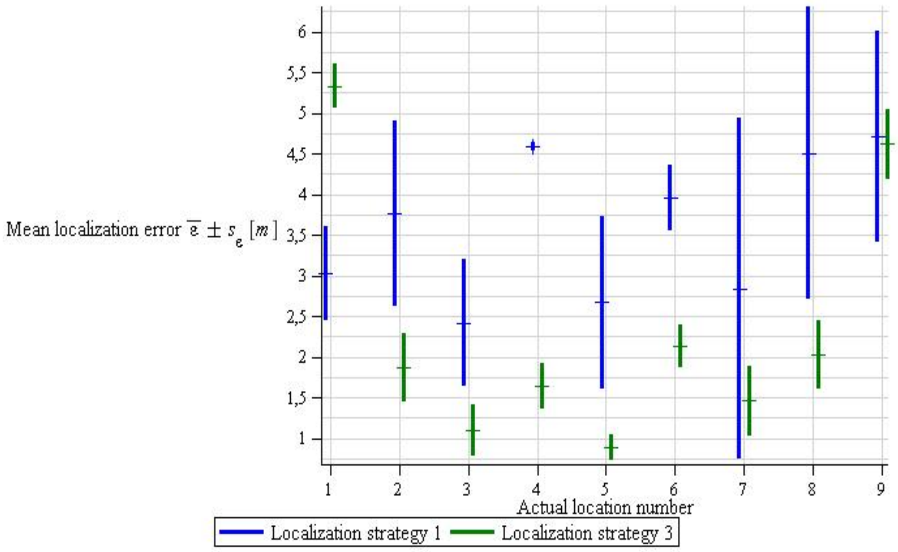


Figure 5.13: Performance of localization strategies 1 and 3 tested in the *forth* session of experiments

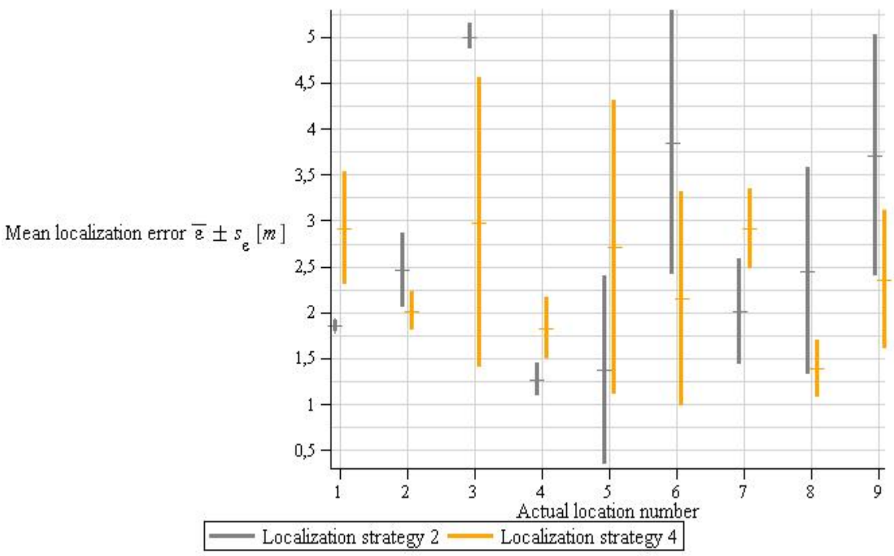


Figure 5.14: Performance of localization strategies 2 and 4 tested in the *fifth* session of experiments

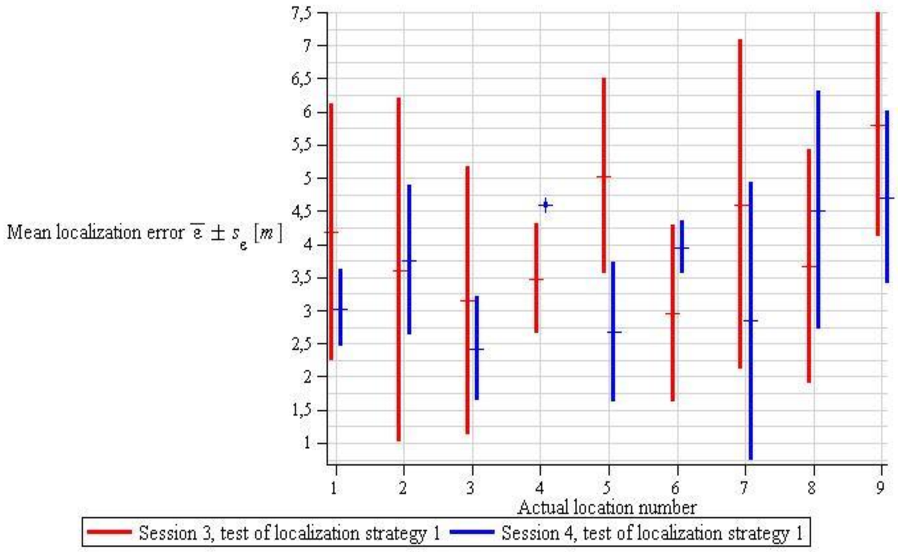


Figure 5.15: Performance of localization strategy 1 tested in the *third* and in the *forth* sessions of experiments

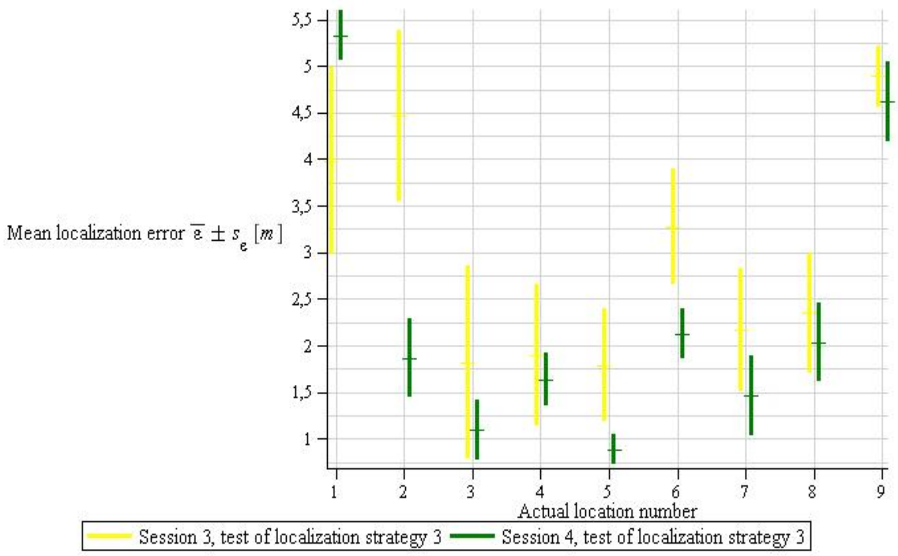


Figure 5.16: Performance of localization strategy 3 tested in the *third* and in the *forth* sessions of experiments

5.3.8.3 Comparison of the localization strategies 1 and 2, and the localization strategies 3 and 4 tested in the *forth* and in the *fifth* sessions of experiments correspondingly

The fifth session of experiments aimed at evaluation of performance of the implemented ILS with its preliminary calibration with the help of an automatic movement (in this case, of a line-following robot). Tests of the localization strategies 2 and 4 conducted in the fifth session of experiments have shown ambiguous results.[14] On the one hand, the test of the localization strategy 2 has demonstrated advantage of preliminary calibration of the system with the help of an automatic movement: localization accuracy of the system for the test venue as a whole has improved in comparison with the same parameter of the localization strategy 1 that had identical nodes' deployment and that was tested in the previous session of experiments (see Figure 5.10). The improvement in localization accuracy is also confirmed by comparison of the named two localization strategies on the level of actual locations of the tag (see Figure 5.17). Regarding deviation of localization errors from their means for the test venue as a whole the named two localization strategies have resulted in almost identical sample standard deviation values (see Figure 5.10).

On the other hand, the test of the localization strategy 4 has not confirmed improvement in localization accuracy of the system detected after the test of the localization strategy 2: the localization strategy 4 has resulted in almost the same localization accuracy for the test venue as a whole as the localization strategy 3 that had identical nodes' deployment and that was tested in the frame of the previous session of experiments (see Figure 5.10). Moreover, localization accuracy of the strategy 3 for the test venue as a whole has been impaired by strong deteriorations in its localization accuracy for the first and for the ninth actual locations of the tag (see Figure 5.18) and should be even better. Regarding deviation of localization errors from their means for the test venue as a whole the localization strategy 4 has resulted in a lower one than the localization strategy 3 (see Figure 5.10). However, this parameter of the localization strategy 3 could be affected by the accuracy fluctuations mentioned above (see Equations 5.6 and 5.11) and cannot be considered for an objective comparison.

In order to confirm positive effect of the preliminary calibration of the system with the help of an automatic movement, repeated sessions of experiments are needed.

[14] See Table D.4 in comparison with Table D.3 provided in appendices.

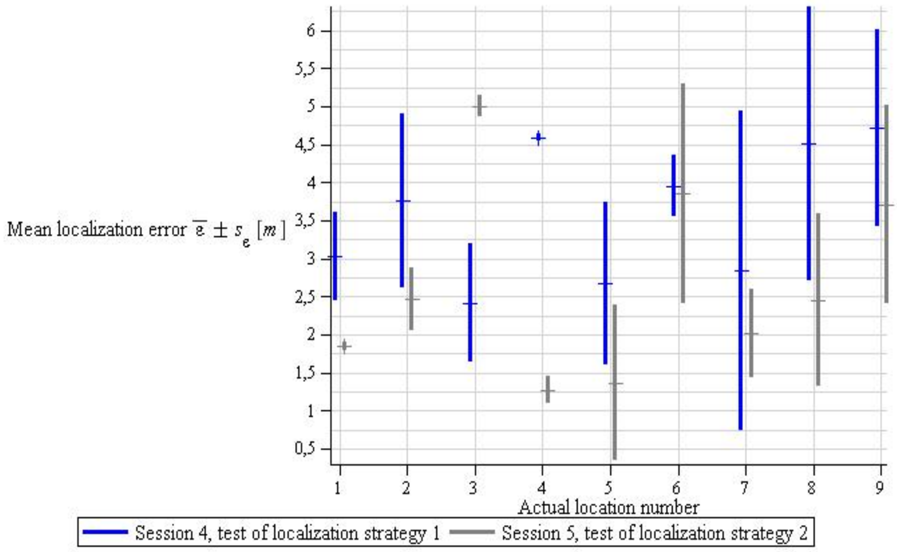


Figure 5.17: Performance of localization strategies 1 and 2 tested in the *forth* and in the *fifth* sessions of experiments correspondingly

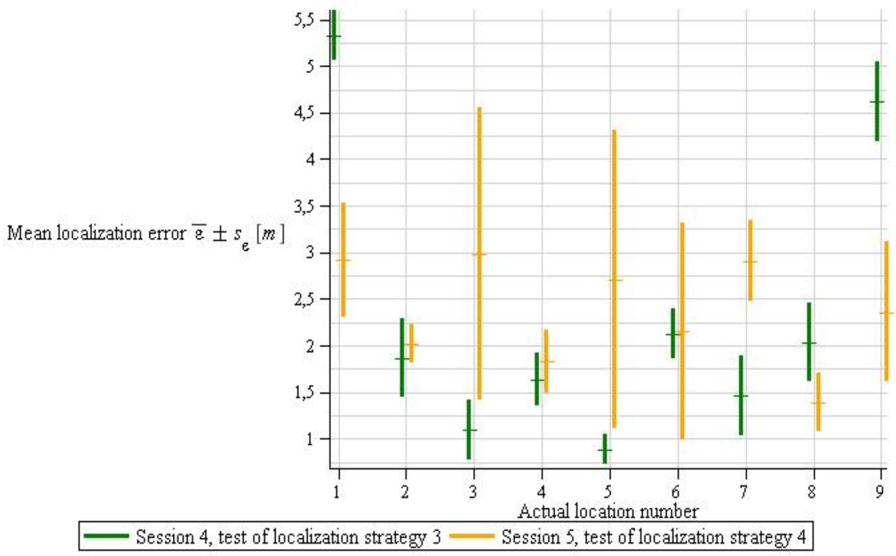


Figure 5.18: Performance of localization strategies 3 and 4 tested in the *forth* and in the *fifth* sessions of experiments correspondingly

6 Conclusions

This chapter starts by summary of the conducted research (see Section 6.1) and is concluded with suggestions for possible extensions of the implemented prototype ILS and with suggestions for future research (see Sections 6.2 and 6.3 correspondingly).

6.1 Summary

Being led by the aim of creating a low-cost ILS that could be used by SMEs an RSSI-based ILS has been proposed for its further implementation. Advantage of the RSSI-based ILSs consists in portability, low cost and low power consumption of RF sensors that this type of systems uses. As an approach for the RSSI-based indoor localization, triangulation that operates with the path-loss nature of RF signals has been proposed. The RSSI-based triangulation is the most promising for building a low-cost ILS, since it can be applied only with few modifications of the base technology and with hardware of standard wireless technologies such as BLE and Wi-Fi.

Usage of standard wireless technology decreases hardware costs and, therefore, corresponds to the purpose of creating a low-cost ILS. As a result, BLE technology that is currently widely spread among modern mobile phones and portable electronic devices and that possesses low cost and low power consumption has been proposed for its usage.

In order to increase its localization accuracy, an RSSI-based ILS demands calibration. The calibration procedure is time-consuming and, consequently, increases cost of the system. For a low-cost ILS efforts associated with its calibration should be reduced or even totally eliminated while providing a localization accuracy that is sufficient for reducing search times on the shop floor, which means that positions should be known at least in a sub-room level.

Two research concepts have been proposed in this master's thesis. The first concept aims at leveraging automatic movement that is available in industry (e.g. a conveyor) for the purpose of simplification of the required calibration procedure. The second concept uses range-free localization technique with fixed nodes that are spatially distributed into emitters and receivers. Range-free localization techniques (unlike the range-based ones) do not demand calibration of the system and, thus, correspond to the concept of a low-cost ILS needed to be designed and implemented. As a localization approach, the core idea of the ROCRSSI range-free localization technique for the first time formulated by Liu et al. [1] has been proposed to use (see description of the ROCRSSI provided in Section 3.2).

Based on the core idea of the ROCRSSI and on the two named research concepts five localization strategies have been proposed (see detailed description of the strategies presented in Section 3.3). In order to test these localization strategies and, thus, investigate the proposed research concepts, an RSSI-based ILS has been implemented.

From the design point of view, the implemented ILS represents a client-server architecture. The clients are represented by BLE compatible Android smartphones and serve as BLE signals' receivers. Communication between the clients and the server deployed on a laptop has been organized with the help of web services via Wi-Fi. The server is intended for allocation of the web services and DB, and for performance of the localization algorithm (see description of the system's architecture provided in detail in Section 3.1 as well as its implementation details provided in Chapter 4).

Performance of the implemented ILS has been evaluated in several sessions of experiments. The experiments have been conducted in two test venues (namely, in the meeting room and in the classroom that is situated in the clean room building). During the experiments four from the proposed five localization strategies (namely, the localization strategies 1–4) have been tested (see description of the test venues as well as review of the conducted sessions of experiments and detailed analysis of the obtained localization results provided in Section 5.3). The obtained localization results have shown the following:[1]

1. Spatially distributed deployments of anchors and reference nodes tested in the conducted sessions of experiments have performed better than the spatially merged ones (both in terms of localization accuracy and in terms of deviation of localization errors from their means). This confirms the assumption about better performance of the former ones made in Section 3.3.3. Such kind of improvement could be explained by increase in variety of rings that are able to be generated by every anchor. This itself enlarges variety of the rings' intersections and shrinks the intersections' sizes. As a result, improvement in localization accuracy of the system is achieved. Variety of rings that could be generated by every anchor can be enlarged even more by increasing number of deployed reference nodes (standard BLE beacons) that are comparatively cheap.
2. High variance of RSSIs led to the high variance of the system's localization accuracy. In order to improve localization accuracy of the system, an RSSI filter has been implemented after the third session of experiments. The filter has, in general, improved localization accuracy of the system and decreased deviation of localization errors from their mean. Implementation of the filter has been complicated by the BLE scan behavior of Motorola Moto G 2014 Android smartphones acting as clients of the implemented ILS. It has been found empirically for the Motorola Moto G 2014 Android smartphones

[1] The results are provided in Tables D.1, D.2, D.3 and D.4, and are summarized in Table D.5 in appendices.

that a short latency of the scan rounds together with a large number of their repetitions provides good compromise between the total amount of time that is spent on the scan and the number of accumulated scan results. However, even such a good compromise has allowed to accumulate only 10–15 scan results for their further filtering after 100 of 200 milliseconds scan rounds (20 seconds in total). Despite the positive effect of the implemented filter on the localization accuracy of the system and on the deviation of localization errors from their mean, the accumulated 10–15 scan results could not be considered sufficient for their further filtering. Accumulation of a greater number of scan results would increase the total time spent on the scan. Theoretical basis of the implemented filter is provided in Section 3.5. Details of its implementation in the client's scan procedure are described in Section 4.2.2.

Significantly better scan behavior is provided, for instance, by BLE smart dongles (e.g. by BLED112). Implementation of the ILS on their basis would allow to accumulate sufficient number of scan results during a shorter period of time.

3. Tests of the localization strategies 2 and 4 that leverage preliminary calibration of the system with the help of an automatic movement (in this particular case, with the help of a line-following robot) have led to ambiguous results: whereas the localization strategy 2 has shown betterment in localization accuracy of the system for the test venue as a whole, the localization strategy 4 – has not.

Among all of the tests of localization strategies conducted in the classroom that is situated in the clean room building (width of the test venue amounts 8.55 m, length – 10.45 m) the best localization accuracy for the test venue as a whole has been achieved by the localization strategy 3 tested in the frame of the fourth session of experiments with filtering of the obtained RSSIs. Although the localization accuracy of the strategy 3 for the test venue as a whole has been impaired by strong deteriorations in its localization accuracy for the first and for the ninth actual locations of the tag (see Figure 5.18), it has come to 2.34 m. Accumulation of a greater number of scan results for their further filtering or use of additional reference nodes might increase the localization accuracy of the system even more.

6.2 Suggestions for further extensions of the system

Recommendations listed below are directed towards extension of the implemented ILS. However, some of them can be considered as general recommendations for an ILS's implementation. The suggestions for further extension of the system are:

1. *Implementation of principles of ROCRSSI++:* The implemented ILS leverages principles of the original ROCRSSI localization technique proposed by Liu et al. [1] and of its first refinement proposed by Crepaldi et al. [13] and denoted by them as ROCRSSI+. However, the ROCRSSI+ possesses some inefficiencies that have been overcome by

the next refinement of the localization technique proposed by Frattini et al. [10] and denoted by them as ROCRSSI++. Refinements that have been proposed by the authors of ROCRSSI++ could also be realized in the implemented ILS. For instance, since distances between anchors and reference nodes are known to the system, it is possible to evaluate reliability of the obtained RSSIs. Greater RSSI that has been obtained from a longer distance should be considered more reliable than a lower one obtained from a shorter distance. The latter is considered to be affected by more obstacles than the former one. In the implemented ILS variability of RSSIs has been smoothed out by the filter accumulating RSSIs during some period of time, filtering out particularly high and low accumulated values and averaging the rest. However, the authors of ROCRSSI++ localization technique propose also to apply a weighted mean for the averaging. Greater RSSIs should have greater weights and, as a result, affect the average RSSI in greater degree, since the greater RSSIs are considered to be more dependent only on distance and less on obstacles. The last refinement that could be realized as an extension of the implemented ILS refers to channel symmetry. Under a channel the authors of ROCRSSI++ understand a communication link between two fixed nodes separated by a known distance. From the two RSSIs obtained by fixed nodes of the channel the greatest one is chosen as a reciprocal RSSI value of the channel. Principle of the channels symmetry could be applied for the localization strategies 1 and 3 operating with spatially merged fixed nodes.

Description of the original ROCRSSI localization technique and its refinements is provided in Section 3.2. Proposed localization strategies are described in Section 3.3.

2. *Usage of running average while filtering:* The implemented filter operates with idea of discrete accumulation of RSSIs during the preset time interval (see description of the realized filter from the theoretical and from the practical points of view in Sections 3.5 and 4.2.2 correspondingly). After the accumulation, particularly high and low RSSIs are filtered out and the rest is averaged. Radius Networks propose to use running average of RSSIs for their filtering [15]. They suggest to filter measurements that have been received during the most recent 20 seconds and do not to wait their accumulation for further filtering. Implementation of such kind of filter would allow to smooth discretization of the localization process. However, as it has been already mentioned in Section 3.5, both of the filters (the original one and the implemented one) deprive the system of its ability to react to any movements of scanning or/and emitting device(s) on-the-fly.

3. *Cloud-based ILS service:* The idea of a cloud-based ILS totally corresponds to the necessity to create an ILS that would offer a simple way of introduction and minimal operational effort required by SMEs. A cloud-based ILS would be able to propose required hardware based on the shop floor plan and further frame conditions and would allow hardware of the ILS being installed and used without the need for configuration by SMEs.

4. *Implementation of a centralized remote control of clients:* The implemented ILS demands controlling the realized scan and fingerprints' acquisition procedures (namely, their start

and stop) locally from every client (see GUI presented and described in Section 4.2.1). If the clients operate in a shop floor and are protected by covers, any operations with them will be inconvenient. As a solution, implementation of a client that would serve as a control panel is suggested. The client could undertake the following functionality: setup of experiments, remote start/stop of the scan and the fingerprints' acquisition procedures on the selected clients, graphical representation of the currently conducted localization. The other clients should undertake the scanning functionality only. The described scenario could be implemented, for example, with the help of the WebSocket protocol providing full-duplex connection that can be established once and kept for a long time.

5. *Localization of several tags:* Currently only a single tag can be localized by the implemented ILS. In order to bring it closer to the reality, where several work pieces and/or tools may require their localization, ability to localize several tags should be implemented.

6. *Usage of several conveyors:* The implemented ILS considers leveraging an automatic movement (e.g. a conveyor) for simplification of calibration procedure. In the implemented ILS sections of conveyor are considered to be connected in series (without gaps). Thus, they constitute a single conveyor. If there are gaps between the sections, several conveyors are considered in the system. In order to bring the implemented ILS closer to the reality, where several conveyors may be situated on the shop floor, ability to acquire fingerprints from several separate conveyors should be added to the system.

7. *Usage of capability of Android 5.0 Lollipop to act as a peripheral BLE device:* Starting from Android 5.0 an Android device is able to act not only as a central (scanning) device, but also as a peripheral (emitting) one [18]. Since all of the clients of the implemented ILS are represented by Motorola Moto G 2014 smartphones controlling by Android 5.0, they also could be used as additional emitting devices (as additional reference points of the system).

8. *Usage of the* $distance(RSSI)$ *relation:* Since anchors of the system obtain some of the RSSIs from reference nodes, which coordinates and, hence, distances to the anchors are known or acquire fingerprints, which coordinates are calculated and known as well, the $distance(RSSI)$ relation could be easily built by the system. Let us assume a test venue, where three fixed nodes FN_1, FN_2 and FN_3 are operating (see Figure 6.1). Let us also assume that the fixed node FN_1 has obtained $RSSI_{TFN_1}$ from the sought tag T. Based on the preliminary built $distance(RSSI)$ relation, the system is able to estimate distance $d_1 = distance(RSSI_{TFN_1})$ from the fixed node FN_1 to the tag T. Thus, a circle, in which proximity, according to the fixed node FN_1, the tag is most probably situated, is determined by the system. Such an estimation of the distance could be used for calculation of CoG of a formed intersection area. Instead of arithmetic mean (average), weighted arithmetic mean is proposed to be used for calculation of the CoG. Weights are proposed to be assigned to the cells of the intersection area in accordance with proximity

of their centers to the determined circles: the closer is the cell's center to the circle, the higher is the cell's weight. Let us again consider the fixed node FN_1 and the circle of radius d_1, in which proximity, according to the fixed node, the tag is situated. In this case, higher weights (m) could be assigned to the cells of the intersection area, which centers lie within the proximity ring with inner and outer radiuses $(d_1 - \Delta d)$ and $(d_1 + \Delta d)$ correspondingly. To all of the other cells of the intersection area lower weights ($n < m$) could be assigned. Value of the distance Δd could be chosen, for instance, based on distances between the fixed nodes and on degree of granulation of the test venue. The procedure of the weights' assignment should be repeated for the rest of two fixed nodes of the system. As a result every cell of the intersection area will have three assigned weights. Based on these assigned weights, total weight should be calculated for every cell of the intersection area. Total weights are substituted to the weighted arithmetic mean formula for calculation of CoG of the intersection area. It is necessary to mention here that distribution of weights within the intersection area may be subordinated to more complex rules. Both the distribution of weights and calculation of their total values should be investigated in more details.

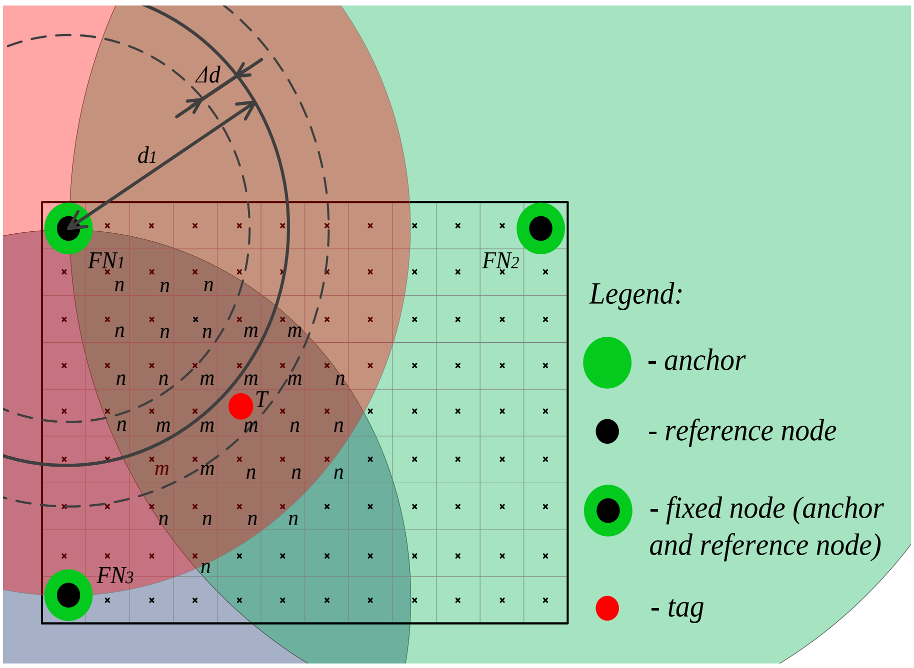


Figure 6.1: Usage of the $distance(RSSI)$ relation for calculation of CoG of an intersection area

6.3 Suggestions for future research

In Section 2.4 two research concepts have been proposed. One of the concepts leverages an automatic movement for simplifying the calibration procedure. Another one leverages the ROCRSSI range-free localization technique operating with fixed nodes that are spatially distributed into emitters and receivers. In the same section research questions have been stated. The questions address impact of the proposed concepts on the localization accuracy of the system. In order to wholly investigate the two proposed research concepts and answer with more confidence the stated research question the localization strategies formulated in Section 3.3 should be tested more thoroughly.

The formulated localization strategies can be tested either with the help of the implemented Android-based ILS or of any newly created system (for instance, of a BLE smart dongle-based ILS). For the thorough test of the localization strategies the following lines of investigations are proposed (description of already conducted experiments is provided in detail in Section 5.3):

1. *Experiments in application center:* The application center of Fraunhofer IPA represents an environment that is very similar to a factory shop floor including a conveyor and machines and that is used for demonstrating and show-casing new technology. The conveyor is composed of separate movable segments that form a U-shape in the middle of the application center. The width of the application center is about 10 m, the length – about 35 m. The application center has its ceiling on the level of ceiling of the second floor (counting from the ground floor). Because of its similarity to a factory shop floor the application center is of interest of conducting experiments. The implemented ILS and its localization strategies should be tested there.
2. *Localization nearby the walls:* While actual locations of the tag being in the middle of the test venues have been examined during the experiments, the area that adjoins the walls of the test venues (around 1.5 m away from the walls in the meeting room and around 2 m away from the walls in the classroom that is situated in the clean room building) has remained uncovered. In order to get a complete picture of the system's localization ability all over the test venues, the area nearby the walls should be examined as well.
3. *Degree of granulation:* During configuration of experiments conducted in this master's research their test venues have always been equally granulated onto the cells. Namely, size of every cell always came to 0.32 m (width) and 0.31 m (length). Degree of granulation (the size of the cells) has been fixed for all of the conducted experiments in order to have a chance to compare their results objectively. Thorough investigation of the proposed localization strategies demands examination of other degrees of granulation. As a result, the degree of granulation providing the best localization accuracy for the particular test venue should be determined.

4. *Schemes of nodes' deployments:* One scheme of spatial deployment of nodes (anchors and reference nodes) may operate better than another one. In this master's research schemes of nodes' deployments were repeated from one session of experiments to another one conducted in the same test venue. Such a constancy has allowed to perform impartial evaluation in the end. However, other schemes of nodes' deployments should be also investigated. For instance, anchors could be distanced a bit farther away from the walls or some of them could be even placed in the middle of the test venue. Since reference nodes of the implemented ILS are represented by BLE beacons that are comparatively cheap, the number of reference nodes can be significantly increased. As a result, the reference nodes could be deployed, for example, in a form of a grid covering the whole test venue.

5. *Number of nodes:* This line of investigations is directed to determine optimal numbers of nodes (anchors and reference nodes) that are able to provide required localization accuracy at test venues of different sizes. Experiments should be conducted for different localization strategies and should consider the degree of granulation and the scheme of nodes' deployment that have shown the best localization accuracy during the previous stages of experiments.

6. *Localization strategies 2 and 4:* In this research the named two localization strategies that leverage an automatic movement for simplifying the system calibration procedure have been tested in the classroom that is situated in the clean room building. The tests have shown ambiguous results (see Sections 5.3.7 and 5.3.8). Besides of testing the strategies in the application center they should be also additionally tested in the classroom that is situated in the clean room building.

7. *Localization strategy 5:* In this research four from the proposed five localization strategies have been tested. The only one localization strategy that has remained untested is the localization strategy 5 that utilizes fingerprints obtained with the help of an automatic movement and anchors only. Since the localization strategy 5 operates with preliminary acquired fingerprints and does not use real time RSSI measurements from reference nodes, the strategy could help to evaluate influence of changes within the test venue (moving humans, equipment facilities placed in a new way etc.) on the localization accuracy of the system.

Appendices

A Definition of terms

Actual location of the tag is a point of a test venue where the **tag** is currently placed. Coordinates of the actual location of the tag are known to the system and are used for comparison with coordinates of **estimated location of the tag** in order to calculate localization error.

Anchor is a BLE compatible Android smartphone serving as signals' receiver. Coordinates of anchors are known to the system and are used for implementation of localization algorithm. In the client-server architecture of the system anchors play role of clients.

Estimated location of the tag represents coordinates of the **tag** calculated by the system.

Fingerprint is a spatially anchored RSSI. Coordinates of fingerprints are calculated during the fingerprints' acquisition procedure and are used for implementation of localization algorithm. Together with **reference nodes**, fingerprints play role of **reference points** of the system.

Fixed node is a generic name of a node that represents a spatial merge of an **anchor** and a **reference node**.

Reference node is a BLE beacon serving as signals' emitter. Coordinates of reference nodes are known to the system and are used for implementation of localization algorithm. Together with **fingerprints**, reference nodes play role of **reference points** of the system.

Reference point is a point of a test venue which coordinates and related RSSI are known to the system. As reference points, **reference nodes** and **fingerprints** are able to serve. Reference points are used for implementation of localization algorithm (namely, for rings' and circles' construction that underlies the exploited ROCRSSI localization technique).

Tag is a standard BLE beacon that plays role of an asset (sought) beacon.

B Data model diagrams

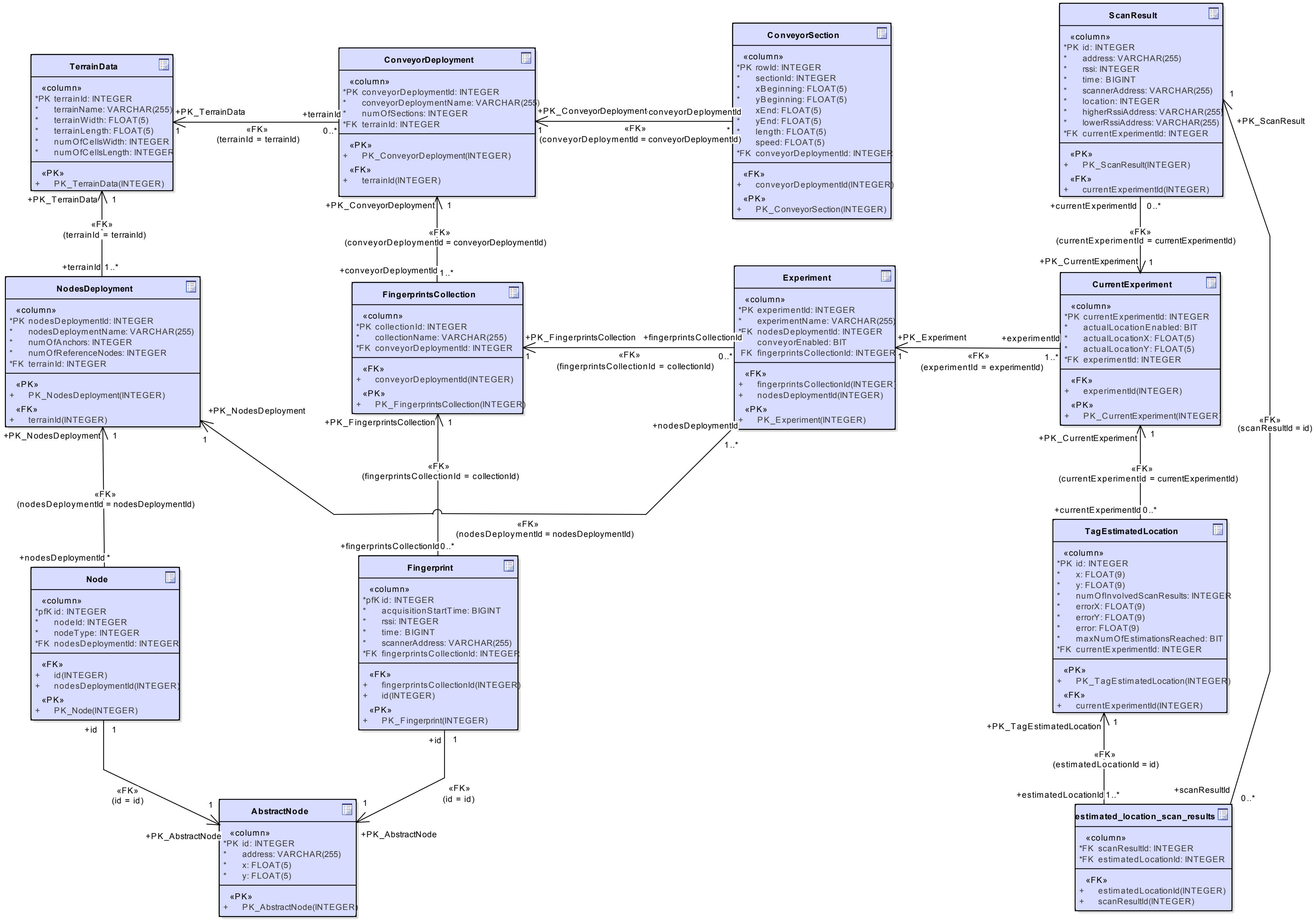


Figure B.1: DMD of the server's DB

C Supplemental UML diagrams

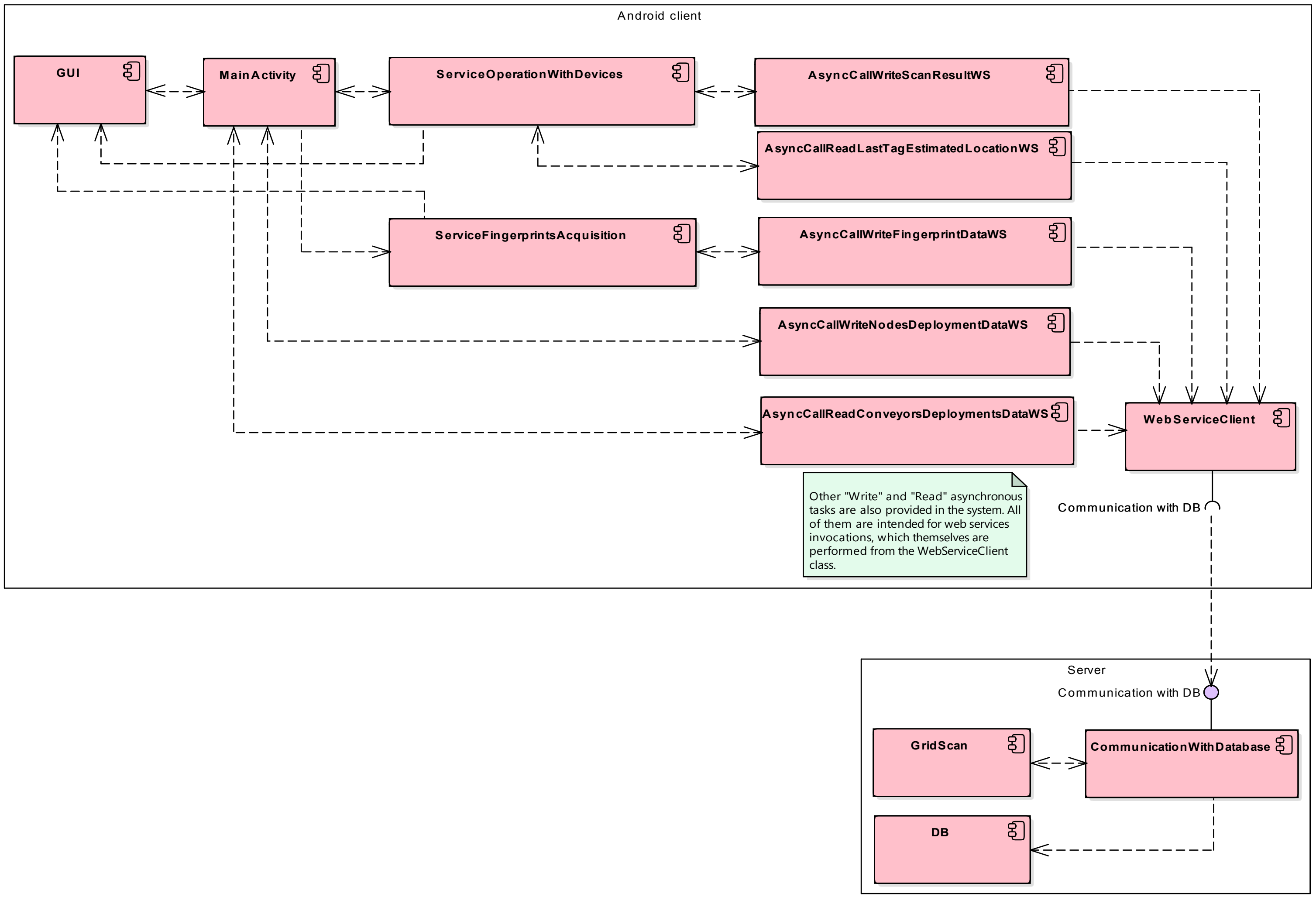


Figure C.1: Overview of components used in the following sequence diagrams

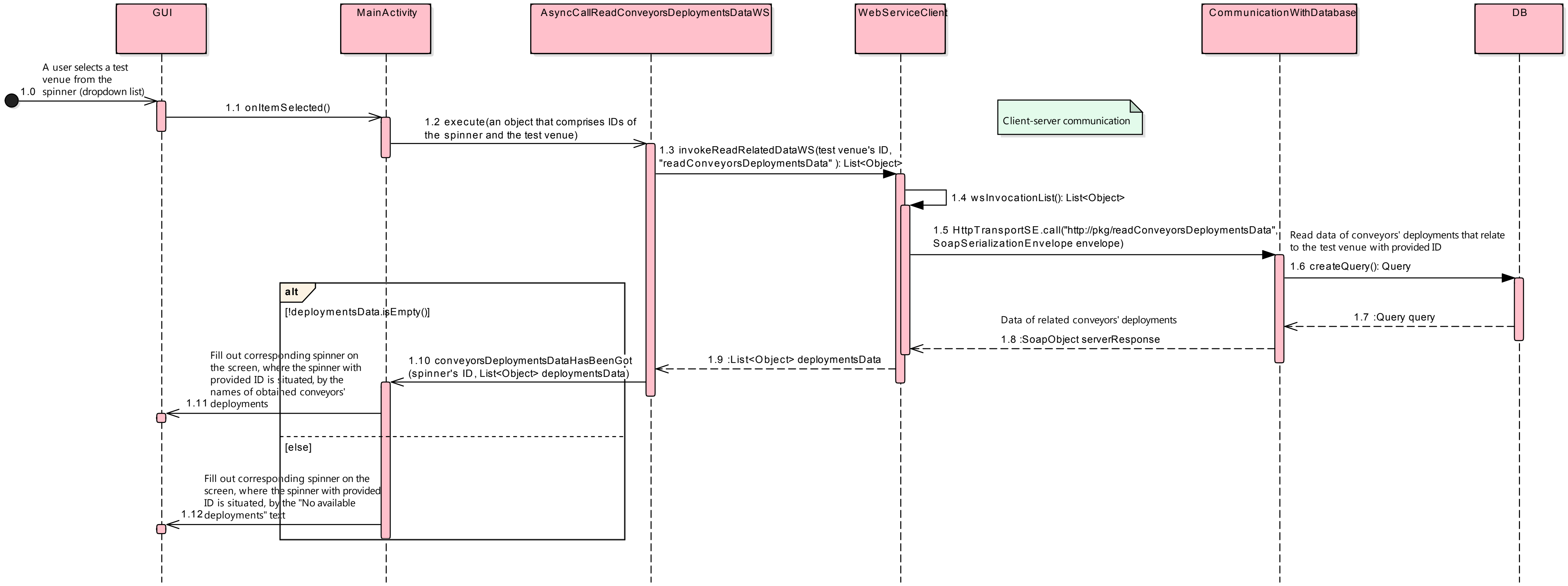


Figure C.2: Sequence diagram illustrating load of GUI data by the example of reading data of conveyors' deployments that relate to the test venue with provided ID

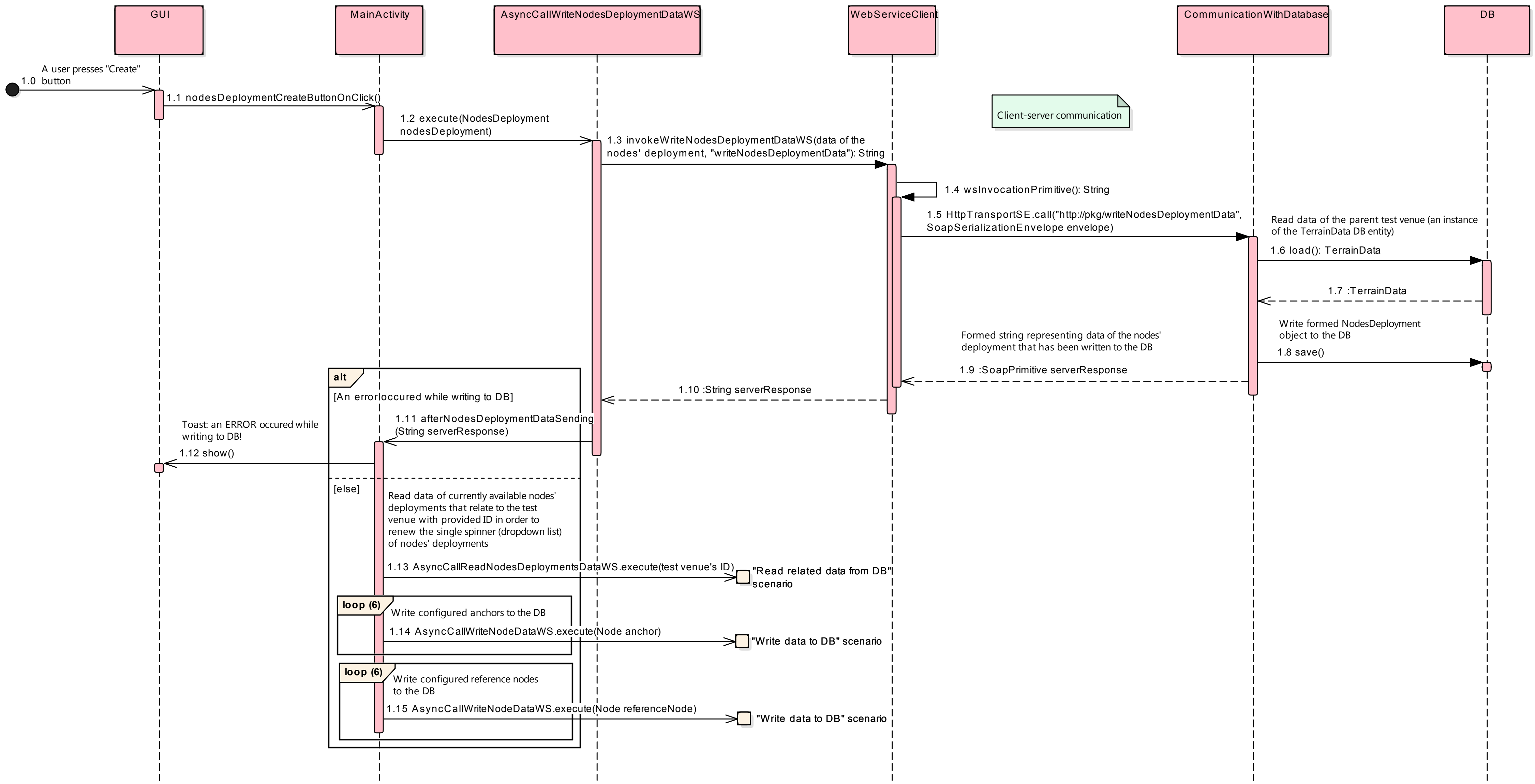


Figure C.3: Sequence diagram illustrating persisting of data by the example of writing data of a nodes' deployment

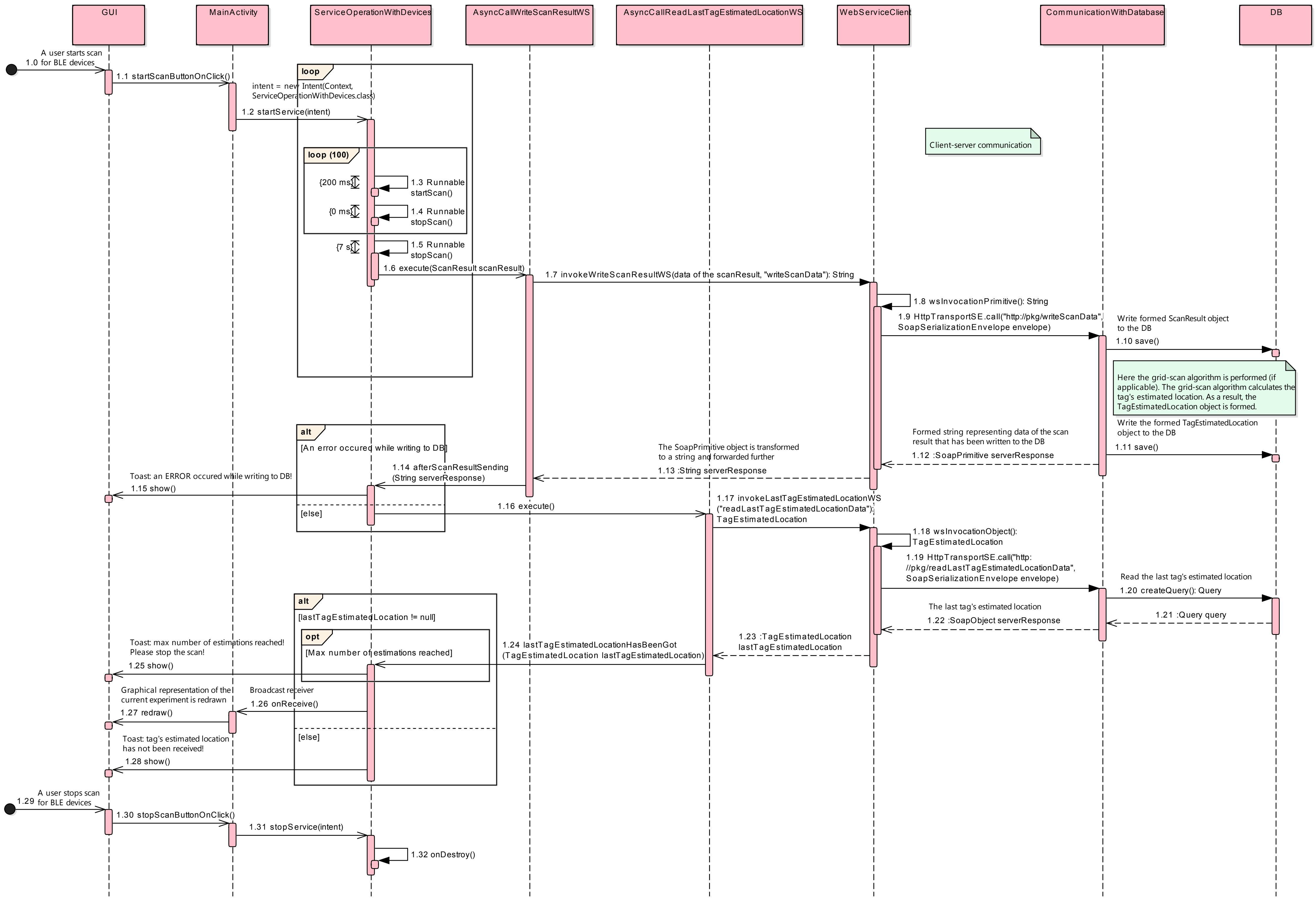


Figure C.4: Sequence diagram illustrating the scan procedure and the subsequent tag's estimated location reading

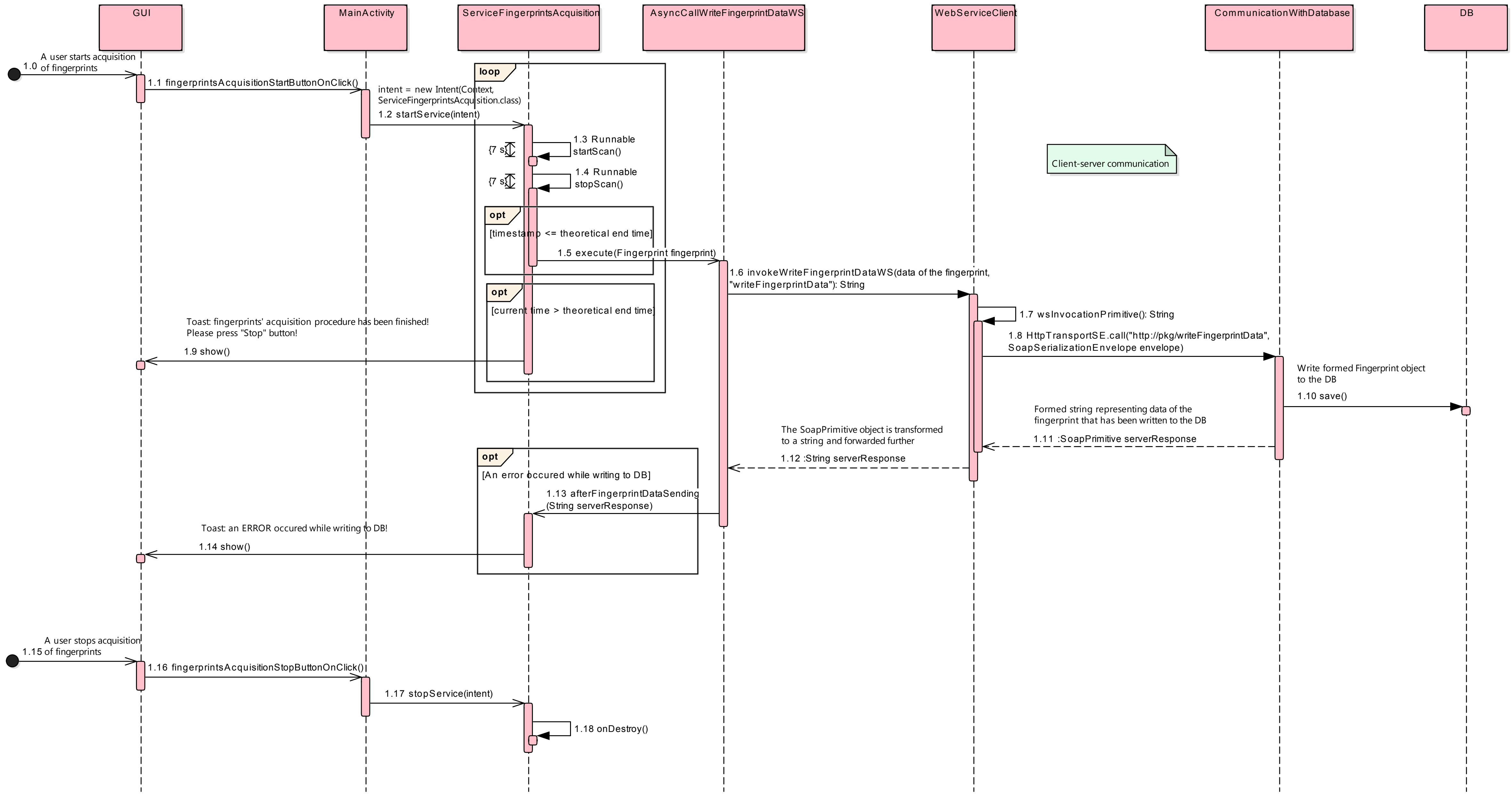


Figure C.5: Sequence diagram illustrating the fingerprints' acquisition procedure

D Localization results

D.1 Session 2

Test venue: meeting room				Actual location of the tag				Estimated location of the tag						
Tested localization strategy	**Maximal error [m]**	**Mean error $\bar{E}$ [m]**	**Sample standard deviation s_E [m]**	**Number**	**x [m]**	**y [m]**	**z [m]**	**Number of estimations**	**Mean absolute x-error $\bar{\varepsilon}_{abs_x}$ [m]**	**Mean absolute y-error $\bar{\varepsilon}_{abs_y}$ [m]**	**Minimal error [m]**	**Maximal error [m]**	**Mean error $\bar{\varepsilon}$ [m]**	**Sample standard deviation s_ε [m]**
				1	1,80	1,00	0,73	50	1,22	1,48	0,29	5,32	1,98	1,21
				2	2,60	1,00	0,73	50	0,62	0,37	0,45	1,12	0,80	0,15
				3	3,40	1,00	0,73	50	0,63	0,90	0,08	3,84	1,14	1,19
				4	1,80	2,60	0,73	50	0,83	1,39	0,70	2,81	1,70	0,62
1	5,32	1,57	0,96	5	2,60	2,60	0,73	50	1,04	1,61	0,58	3,55	2,05	0,67
				6	3,40	2,60	0,73	50	0,75	1,30	0,16	2,37	1,56	0,57
				7	1,80	4,20	0,73	50	0,38	0,65	0,26	1,92	0,82	0,32
				8	2,60	4,20	0,73	50	1,22	1,24	0,15	3,97	1,79	0,94
				9	3,40	4,20	0,73	50	1,45	1,55	0,19	4,29	2,26	1,06
				1	1,80	1,00	0,73	50	0,88	1,52	0,32	3,30	1,85	0,85
				2	2,60	1,00	0,73	50	0,44	1,62	1,14	4,03	1,71	0,53
				3	3,40	1,00	0,73	50	0,55	1,70	1,47	2,77	1,82	0,34
				4	1,80	2,60	0,73	50	0,47	0,67	0,21	2,60	0,96	0,55
3	4,03	1,43	0,66	5	2,60	2,60	0,73	50	0,73	0,70	0,23	2,23	1,11	0,49
				6	3,40	2,60	0,73	50	0,38	0,43	0,08	1,21	0,64	0,27
				7	1,80	4,20	0,73	50	0,36	1,59	0,72	2,31	1,66	0,35
				8	2,60	4,20	0,73	50	0,28	1,30	0,34	2,31	1,34	0,46
				9	3,40	4,20	0,73	50	1,30	0,99	1,05	3,66	1,77	0,57

Table D.1: Localization results of session 2

D.2 Session 3

Test venue: classroom in the clean room building				Actual location of the tag				Estimated location of the tag						
Tested localization strategy	Maximal error [m]	Mean error $\bar{E}$ [m]	Sample standard deviation s_E [m]	Number	x [m]	y [m]	z [m]	Number of estimations	Mean absolute x-error $\bar{\varepsilon}_{abs_x}$ [m]	Mean absolute y-error $\bar{\varepsilon}_{abs_y}$ [m]	Minimal error [m]	Maximal error [m]	Mean error $\bar{\varepsilon}$ [m]	Sample standard deviation s_ε [m]
				1	6,25	8,55	0,73	90	2,32	3,11	0,19	7,82	4,18	1,93
				2	4,15	8,55	0,73	90	1,21	3,27	0,69	8,99	3,61	2,60
				3	2,05	8,55	0,73	100	1,22	2,69	0,26	7,06	3,15	2,02
				4	6,25	5,75	0,73	100	2,61	1,92	1,94	5,13	3,47	0,83
1	9,19	4,05	2,05	5	4,15	5,75	0,00	100	2,45	4,29	1,05	6,87	5,03	1,48
				6	2,05	5,75	0,73	100	1,62	2,18	0,40	6,95	2,95	1,34
				7	6,25	2,95	0,73	100	1,99	3,87	0,68	9,19	4,60	2,49
				8	4,15	2,95	0,00	100	2,32	2,65	0,37	7,89	3,66	1,76
				9	2,05	2,95	0,73	100	4,29	3,68	2,47	8,68	5,81	1,69
				1	6,25	8,55	0,73	50	3,28	1,98	2,06	5,90	3,98	1,01
				2	4,15	8,55	0,73	50	0,81	4,28	2,85	6,15	4,46	0,92
				3	2,05	8,55	0,73	50	0,83	1,46	0,12	3,67	1,82	1,04
				4	6,25	5,75	0,73	50	1,60	0,86	0,91	3,71	1,90	0,76
3	6,15	2,96	1,38	5	4,15	5,75	0,00	50	1,64	0,58	0,63	3,08	1,79	0,60
				6	2,05	5,75	0,73	50	3,19	0,51	2,28	4,31	3,27	0,62
				7	6,25	2,95	0,73	50	1,77	1,08	0,80	3,52	2,17	0,66
				8	4,15	2,95	0,00	50	1,13	2,03	1,01	3,80	2,35	0,65
				9	2,05	2,95	0,73	50	4,14	2,46	4,29	5,34	4,89	0,33

Table D.2: Localization results of session 3

D.3 Session 4

Test venue: classroom in the clean room building				Actual location of the tag				Estimated location of the tag						
Tested localization strategy	**Maximal error [m]**	**Mean error $\bar{E}$ [m]**	**Sample standard deviation s_E [m]**	**Number**	**x [m]**	**y [m]**	**z [m]**	**Number of estimations**	**Mean absolute x-error $\bar{\varepsilon}_{abs_x}$ [m]**	**Mean absolute y-error $\bar{\varepsilon}_{abs_y}$ [m]**	**Minimal error [m]**	**Maximal error [m]**	**Mean error $\bar{\varepsilon}$ [m]**	**Sample standard deviation s_ε [m]**
				1	6,25	8,55	0,73	100	0,75	2,86	2,31	3,49	3,03	0,58
				2	4,15	8,55	0,73	100	2,27	2,51	1,73	6,63	3,76	1,14
				3	2,05	8,55	0,73	100	1,61	1,62	1,88	4,31	2,42	0,78
				4	6,25	5,75	0,73	100	3,65	2,70	4,49	4,65	4,59	0,05
1	7,44	3,61	1,46	5	4,15	5,75	0,00	100	1,15	2,34	1,16	4,20	2,67	1,06
				6	2,05	5,75	0,73	100	3,12	2,07	3,07	4,64	3,95	0,40
				7	6,25	2,95	0,73	100	2,18	1,52	0,70	6,56	2,84	2,10
				8	4,15	2,95	0,00	100	1,59	4,05	2,58	7,44	4,51	1,80
				9	2,05	2,95	0,73	100	4,46	0,86	2,84	6,10	4,71	1,30
				1	6,25	8,55	0,73	100	4,67	2,37	4,67	5,66	5,33	0,27
				2	4,15	8,55	0,73	100	1,08	1,11	1,47	2,70	1,87	0,42
				3	2,05	8,55	0,73	100	0,81	0,67	0,26	1,68	1,09	0,32
				4	6,25	5,75	0,73	100	0,92	0,96	1,19	1,95	1,64	0,28
3	5,66	2,34	1,51	5	4,15	5,75	0,00	100	0,45	0,74	0,74	1,49	0,89	0,16
				6	2,05	5,75	0,73	100	1,90	0,87	1,87	2,89	2,13	0,27
				7	6,25	2,95	0,73	100	0,20	1,43	0,70	1,78	1,46	0,43
				8	4,15	2,95	0,00	100	1,94	0,35	0,89	2,41	2,03	0,42
				9	2,05	2,95	0,73	100	4,12	2,06	2,26	4,82	4,62	0,43

Table D.3: Localization results of session 4

D.4 Session 5

Test venue: classroom in the clean room building				Actual location of the tag				Estimated location of the tag						
Tested localization strategy	**Maximal error [m]**	**Mean error $\bar{E}$ [m]**	**Sample standard deviation s_E [m]**	**Number**	**x [m]**	**y [m]**	**z [m]**	**Number of estimations**	**Mean absolute x-error $\bar{\varepsilon}_{abs_x}$ [m]**	**Mean absolute y-error $\bar{\varepsilon}_{abs_y}$ [m]**	**Minimal error [m]**	**Maximal error [m]**	**Mean error $\bar{\varepsilon}$ [m]**	**Sample standard deviation s_ε [m]**
				1	6,25	8,55	0,73	100	1,83	0,18	1,79	2,14	1,85	0,06
				2	4,15	8,55	0,73	100	2,24	0,79	1,69	4,23	2,46	0,41
				3	2,05	8,55	0,73	100	4,61	1,95	4,76	5,22	5,00	0,14
				4	6,25	5,75	0,73	100	1,15	0,47	0,99	1,39	1,27	0,18
2	7,51	2,67	1,47	5	4,15	5,75	0,00	100	0,70	1,13	0,24	3,05	1,37	1,02
				6	2,05	5,75	0,73	100	3,18	1,91	1,44	7,51	3,85	1,44
				7	6,25	2,95	0,73	100	0,54	1,93	1,22	3,85	2,01	0,58
				8	4,15	2,95	0,00	100	1,43	1,87	0,33	4,05	2,45	1,13
				9	2,05	2,95	0,73	100	3,45	0,84	1,07	7,09	3,71	1,31
				1	6,25	8,55	0,73	100	2,77	0,84	2,35	5,11	2,92	0,61
				2	4,15	8,55	0,73	100	0,66	1,78	1,36	2,35	2,02	0,21
				3	2,05	8,55	0,73	100	0,59	2,88	0,34	5,92	2,98	1,57
				4	6,25	5,75	0,73	100	1,24	1,01	0,69	2,50	1,83	0,34
4	5,97	2,36	1,07	5	4,15	5,75	0,00	100	1,61	2,18	0,09	4,34	2,71	1,60
				6	2,05	5,75	0,73	100	1,41	1,18	0,96	5,97	2,15	1,16
				7	6,25	2,95	0,73	100	2,29	1,70	1,74	3,58	2,91	0,44
				8	4,15	2,95	0,00	100	0,88	0,97	1,15	2,71	1,39	0,31
				9	2,05	2,95	0,73	100	0,69	2,01	1,23	3,42	2,36	0,75

Table D.4: Localization results of session 5

D.5 Summary table of localization results

Session of experiments	Test venue	Tested localization strategy	Maximal error [m]	Mean error $\bar{E}$ [m]	Sample standard deviation s_E [m]
Session 2	meeting room	1	5,32	1,57	0,96
		3	4,03	1,43	0,66
Session 3	classroom in the clean room building	1	9,19	4,05	2,05
		3	6,15	2,96	1,38
Session 4	classroom in the clean room building	1	7,44	3,61	1,46
		3	5,66	2,34	1,51
Session 5	classroom in the clean room building	2	7,51	2,67	1,47
		4	5,97	2,36	1,07

Table D.5: Localization results summary

Bibliography

[1] C. L. C. Liu, K. W. K. Wu, and T. H. T. He, "Sensor localization with Ring Overlapping based on Comparison of Received Signal Strength Indicator," *2004 IEEE International Conference on Mobile Ad-hoc and Sensor Systems (IEEE Cat. No.04EX975)*, 2004. (Cited on pages 17, 23, 26, 27, 28, 31, 104 and 105)

[2] P. Hawkins, "Nanochip Fab Solutions, July 2009 - Changing the game in material movement." Available at `http://www.nxtbook.com/nxtbooks/cmp/nanochipfabsolutions_200907/#/4`. (Cited on pages 17 and 18)

[3] E. Akeila, Z. Salcic, A. Swain, A. Croft, and J. Stott, "Bluetooth-based indoor positioning with fuzzy based dynamic calibration," *TENCON 2010 - 2010 IEEE Region 10 Conference*, 2010. (Cited on pages 18, 19 and 41)

[4] "Bluetooth technology website, Press releases - Bluetooth SIG opens qualification program for Bluetooth core specification version 4.0." Available at `http://www.bluetooth.com/Pages/Press-Releases-Detail.aspx?ItemID=106`. (Cited on page 19)

[5] Y. Wang, S. Susheng, X. Yang, and A. Ma, "Bluetooth Indoor Positioning using RSSI and Least Square Estimation," in *IEEE ICFCC*, 2010. (Cited on page 19)

[6] X. Luo, W. J. O'Brien, and C. L. Julien, "Comparative evaluation of Received Signal-Strength Index (RSSI) based indoor localization techniques for construction jobsites," *Advanced Engineering Informatics*, vol. 25, pp. 355–363, 2011. (Cited on page 19)

[7] P. Bahl and V. Padmanabhan, "RADAR: an in-building RF-based user location and tracking system," *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No.00CH37064)*, vol. 2, 2000. (Cited on page 19)

[8] O. S. Oguejiofor, V. N. Okorogu, A. Adewale, and B. O. Osuesu, "Outdoor Localization System Using RSSI Measurement of Wireless Sensor Network," *International Journal of Innovative Technology and Exploring Engineering*, vol. 2, no. 2, pp. 1–6, 2013. (Cited on page 19)

[9] A. Narzullaev and Y. Park, "Novel calibration algorithm for received signal strength based indoor real-time locating systems," *AEU - International Journal of Electronics and Communications*, vol. 67, pp. 637–644, 2013. (Cited on pages 19 and 20)

[10] F. Frattini, C. Esposito, and S. Russo, "Efficient Range-free RSSI Localization Technique for Wireless Sensor Networks," in *4th International workshop on Adaptive and DependAble Mobile Ubiquitous Systems (ADAMUS 2010) - 7th ACM International Conference on Pervasive Services (ICPS 2010)*, 2010. (Cited on pages 19, 23, 26, 28, 29, 30, 32 and 106)

[11] C. Liu and K. Wu, "Performance evaluation of range-free localization methods for wireless sensor networks," *PCCC 2005. 24th IEEE International Performance, Computing, and Communications Conference, 2005.*, 2005. (Cited on page 19)

[12] T. He, C. Huang, B. M. Blum, J. a. Stankovic, and T. Abdelzaher, "Range-free localization schemes for large scale sensor networks," *Proceedings of the 9th annual international conference on Mobile computing and networking MobiCom 03*, p. 81, 2003. (Cited on pages 20, 26 and 30)

[13] R. Crepaldi, P. Casari, A. Zanella, and M. Zorzi, "Testbed implementation and refinement of a range-based localization algorithm for wireless sensor networks," in *Proceedings of the 3rd international conference on Mobile technology applications systems Mobility 06*, p. 61, 2006. (Cited on pages 23, 26, 28, 29 and 105)

[14] G. Zhou, T. He, S. Krishnamurthy, and J. A. Stankovic, "Impact of radio irregularity on wireless sensor networks," in *Proceedings of the 2nd international conference on Mobile systems, applications, and services*, p. 125, 2004. (Cited on page 27)

[15] "Radius Networks, Blog - Fundamentals of beacon ranging," 2014. Available at `http://developer.radiusnetworks.com/2014/12/04/fundamentals-of-beacon-ranging.html`. (Cited on pages 41, 42 and 106)

[16] "Texas Instruments website - CC2541 2.4-GHz Bluetooth low energy and proprietary system-on-chip." Available at `http://www.ti.com/lit/ds/symlink/cc2541.pdf`. (Cited on page 43)

[17] N. Gupta, *Inside Bluetooth low energy*. Artech House, 2013. (Cited on page 43)

[18] "Android developers website, Android 5.0 APIs - Description of intrinsic peculiarities of Bluetooth low energy API." Available at `https://developer.android.com/about/versions/android-5.0.html`. (Cited on pages 52 and 107)

[19] "Android developers website, Android 5.0 APIs - android.bluetooth.le: BluetoothLeScanner." Available at `https://developer.android.com/reference/android/bluetooth/le/BluetoothLeScanner.html`. (Cited on page 53)

[20] "Android developers website, Android 5.0 APIs - android.bluetooth.le: ScanSettings.Builder." Available at `https://developer.android.com/reference/android/bluetooth/le/ScanSettings.Builder.html`. (Cited on page 53)

[21] "Android developers website, Android 5.0 APIs - android.bluetooth.le: ScanSettings." Available at https://developer.android.com/reference/android/bluetooth/le/ScanSettings.html. (Cited on page 53)

[22] "Android developers website - android.os: AsyncTask." Available at http://developer.android.com/reference/android/os/AsyncTask.html. (Cited on page 67)

[23] D. A. S. Fraser, *Statistics. An introduction*. New York: John Wiley & Sons, Inc., 1958. (Cited on page 75)

All links were last followed on July 21, 2015.

Acronyms

AOA angle of arrival. 19

API application programming interface. 45, 52, 53

APIT approximate point-in-triangulation test. 19

BLE Bluetooth low energy. 5, 13, 18, 19, 20, 23, 24, 36, 41, 42, 43, 45, 48, 50, 52, 53, 55, 57, 58, 59, 62, 70, 71, 82, 88, 103, 104, 105, 107, 109, 113

CoG center of gravity. 26, 27, 28, 29, 30, 31, 34, 38, 62, 63, 107

DB database. 15, 23, 24, 25, 34, 39, 41, 45, 46, 48, 60, 61, 62, 63, 67, 68, 69, 70, 71, 104

DMD data model diagram. 25, 60, 62

GUI graphical user interface. 15, 24, 41, 45, 46, 48, 50, 52, 54, 58, 59, 62, 67, 68, 83, 106

ID identifier. 67, 68

ILS indoor locating system. 5, 13, 14, 15, 17, 18, 20, 21, 23, 24, 26, 32, 33, 42, 43, 45, 50, 52, 57, 60, 67, 73, 78, 81, 94, 95, 97, 103, 104, 105, 106, 107, 109

IR infrared. 13

LED light-emitting diode. 17

MES manufacturing execution system. 17, 18

RF radio frequency. 13, 18, 19, 41, 43, 103

ROCRSSI ring overlapping based on comparison of received signal strength indicators. 5, 14, 17, 19, 20, 21, 23, 26, 27, 28, 29, 30, 31, 33, 34, 39, 55, 103, 104, 105, 106, 108, 113

RSSI received signal strength indicator. 5, 13, 14, 15, 19, 24, 26, 28, 29, 30, 31, 33, 34, 36, 39, 40, 41, 42, 43, 46, 55, 57, 58, 59, 60, 61, 70, 71, 78, 79, 85, 88, 89, 94, 97, 103, 104, 105, 106, 107, 110, 113

RTLS real-time locating system. 17, 18

SIG Special Interest Group. 18

SME small and medium enterprise. 5, 13, 103, 106

SNR signal-to-noise ratio. 41, 42

SOAP Simple Object Access Protocol. 67, 68, 69

TI Texas Instruments. 24, 43, 57, 79, 82, 86

TOA time of arrival. 19

UI user interface. 67

UML Unified Modeling Language. 15

WSN wireless sensor network. 19

kNN k-nearest neighbor. 19

Declaration

I hereby declare that the work presented in this thesis is entirely my own and that I did not use any other sources and references than the listed ones. I have marked all direct or indirect statements from other sources contained therein as quotations. Neither this work nor significant parts of it were part of another examination procedure. I have not published this work in whole or in part before. The electronic copy is consistent with all submitted copies.

place, date, signature